Ein adaptives Differenzenverfahren für Konvektions-Diffusionsgleichungen

Dissertation
zur Erlangung des akademischen Grades eines
Doktor der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt im März 2006 im
Fachbereich Mathematik
der Universität Duisburg-Essen, Campus Essen von
Dipl.-Math. Thomas Schamberg aus Essen

Referent: Prof. Dr. rer. nat. W. Heinrichs
Korreferent: Prof. Dr. rer. nat. K. Witsch

Tag der mündlichen Prüfung: 05.07.2006

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

Danksagung

Zu Beginn möchte ich mich bei allen bedanken, die zum Fortgang und Gelingen dieser Arbeit beigetragen haben.
Mein besonderer Dank gilt dabei Herrn Prof. Dr. W. Heinrichs, der diese Arbeit angeregt und die gesamte Zeit über betreut hat. Als Mitarbeiter der Arbeitsgruppe Ingenieurmathematik habe ich von ihm jede nur erdenkliche Unterstützung und Förderung bekommen.
Mein Dank gilt auch Herrn Prof. Dr. K. Witsch, der die Übernahme des Korreferats bereitwillig angenommen hat.
Für die zahllosen anregenden Gespräche möchte ich meinem Bürokollegen Oliver Rheinbach danken. Durch ihn wurde bewiesen, dass sich viele mathematische Probleme lösen, indem man sie jemandem erklärt.
Zum Schluss möchte ich mich bei meiner Familie und meinen Freunden bedanken, die mich während der gesamten Zeit unterstützt haben und ohne die diese Arbeit nicht möglich gewesen wäre.

Kapitel 0

Einleitung

Kovektions-Diffusions-Gleichungen beschreiben Prozesse, in denen Konvektion mit Diffusion gekoppelt ist. Man stelle sich als Beispiel einen Fluss mit starker, gleichmäßiger Strömung vor. In dieses Gewässer wird nun ein Fass mit Öl gegeben. Dabei treten zwei Prozesse auf: Die Verschmutzung diffundiert langsam durch das Wasser und wird gleichzeitig von der Strömung fortgetragen. Der Transportvorgang stellt hierbei den dominanten Part dar. Koppelt man beide Prozesse, so erhält man ein Konvektions-Diffusions-Problem. Das einfachste mathematische Modell eines solchen Prozesses ist das Randwertproblem der Form

$$-\varepsilon u''(x) + a(x)u'(x) + b(x)u(x) = f(x), \quad \text{für } x \in (0,1), \tag{1}$$

mit $u(0) = u(1) = 0$. ε ist hierbei ein kleiner positiver Parameter und a, b und f sind gegebene Funktionen. Die Diffusion wird durch u'' modelliert, die Konvektion durch u'. Die Funktionen u und f nehmen die Rollen einer Quelle bzw. von Transportkräften ein.

Die gesuchte Lösung u dieser Gleichung strebt einem unstetigen Grenzwert entgegen, wenn $\varepsilon \to 0$. Gleichungen dieser Form werden als *singulär gestört* bezeichnet. Die Lösung solcher singulär gestörten Probleme enthält typischerweise Randschichten (*Layer*). Ludwig Prandtl führte 1904 den Begriff des *Randlayers* auf dem dritten internationalen Mathematikerkongress in Heidelberg ein. Man kann einen Layer als sehr kleinen Bereich charakterisieren, in dem die Lösung stark ihr Verhalten ändert.

Singulär gestörte Differentialgleichungen tauchen in vielen Bereichen der angewandten Mathematik auf. Neben der Anwendung im fluiden Fluss verwen-

det man sie in der Simulation des Transportes von Luft, der Grundwasserverschmutzung, der Modellierung von Halbleitern und als Teil der Navier-Stokes Gleichung.

Die Komplexität der dabei auftretenden Gleichungen lassen eine analytische Berechnung der Lösung häufig nur für Spezialfälle zu. Numerische Berechungen sind deshalb von nicht wegzudenkender Wichtigkeit. Bereits in den 50er Jahren des 20. Jahrhunderts wurde die Methode der Finiten Differenzen entwickelt. Das Prinzip hierbei beruht auf der punktweisen Diskretisierung der auftretenden Differentialoperatoren mittels Taylorentwicklung. In den 60er Jahren kam eine weitere lokale Methode hinzu, die Methode der Finiten Elemente. Anders als bei Finiten Differenzen wird hierbei der Raum diskretisiert, indem man elementweise Basisfunktionen entwickelt und die variationelle Formulierung der Differentialgleichung verwendet. Die letzte der drei wichtigen numerischen Methoden kam in den 70er Jahren hinzu, die sogenannten Spektralen Verfahren. Hierbei handelt es sich im Gegensatz zu den beiden anderen um ein globales Verfahren, bei dem mit globalen Ansatzfunktionen gearbeitet wird.

Während Spektrale Verfahren vor allem auf Gebieten mit einfachen Geometrien ihre Anwendung finden, verwendet man die Methode der Finiten Elemente auf komplexeren Geometrien. Im Rahmen dieser Arbeit werden wir untersuchen und zeigen, dass dies auch ohne Weiteres für die Methode der Finiten Differenzen möglich ist und dass man mit ihr beliebig komplizierte Geometrien behandeln kann.

Wann immer man eine numerische Methode auf eine parameterabhängige Gleichung anwendet, hängt das Verhalten der Diskretisierung von diesem Parameter ab. Für singulär gestörte Probleme führen konventionelle Ansätze häufig zu Diskretisierungen, die wertlos sind, wenn der Parameter einen kritischen Wert erreicht. Gerade dieser Wert ist aber in der Regel von besonderem Interesse. Johnson, Rannacher und Boman [JRB95] beobachteten für den Fall der inkompressiblen Navier-Stokes-Gleichung, dass existierende Analysen häufig Konstanten enthalten, welche von $\exp(R)$ (R ist hierbei die Reynoldszahl) abhängen. Sie schlussfolgerten daraus, dass für den Großteil der für die Anwendung interessanten Fälle existierende Fehlerabschätzungen bedeutungslos sind. Dies verdeutlicht die Wichtigkeit der Forderung nach robusten Diskretisierungen und realistischen Fehlerabschätzungen.

Das erste Kapitel dieser Arbeit befasst sich mit den Grundlagen zur Theorie

der partiellen Differentialgleichungen und der Methode der Finiten Differenzen. Es wird ein stabiles Diskretisierungsschema auf der Basis der Upwind-Diskretisierung entwickelt und auf die Konvektions-Diffusions-Gleichung angewendet.

Kapitel 2 behandelt die adaptive Verfeinerung eines gegebenen Gitters $\Omega_h \subset \Omega \subset \mathbb{R}^2$. Dabei wird ein besonderes Augenmerk auf die algorithmische Umsetzung sowie speichertechnische Belange gerichtet.

Die Entwicklung und Herleitung eines geeigneten Fehlerschätzers ist Bestandteil des Kapitels 3. Dabei handelt es sich um einen gewichteten Residuen-Fehlerschätzer, der für den eindimensionalen Fall hergeleitet wird und anschließend auf den zweidimensionalen Fall übertragen wird. Ein weiterer Fehlerschätzer von Axelsson und Nikolova [AN98] wird am Ende des Kapitels vorgestellt.

Kapitel 4 befasst sich mit der Lösung der auftretenden Gleichungssysteme. Dazu werden im Wesentlichen zwei Verfahren verwendet: GMRES und UMFPACK. Beide Löser werden hier ausführlich vorgestellt. Ebenfalls Bestandteil dieses Kapitels ist die Herleitung und Entwicklung eines Postprocessings, welches auf dem adaptiv verfeinerten Gitter die Genauigkeit der gesuchten Lösung verbessern soll.

Im letzten Kapitel wird das entwickelte adaptive Differenzenverfahren auf verschiedene Arten von Layern (Rand-, Ecken- und innere Layer) angewendet, und die Ergebnisse werden diskutiert.

Kapitel 1

Grundlagen

In diesem Kapitel wollen wir das lineare singulär gestörte Randwertproblem

$$\left.\begin{array}{rcll} Lu := -\varepsilon\Delta u + b(x)\cdot\nabla u + c(x)u & = & f(x) & \text{für } x \in \Omega \subset \mathbb{R}^n\ , \\ Bu & = & 0 & \text{auf } \partial\Omega = \Gamma \end{array}\right\} \tag{1.1}$$

betrachten. B ist dabei ein Operator, welcher die Randbedingungen repräsentiert. Ω ist beschränkt und $0 < \varepsilon << 1$. Der Term $-\varepsilon\Delta u$ stellt den Diffusionsanteil dar, $b \cdot \nabla u$ (für $b \neq 0$) den Konvektionsanteil. Die Gesamtheit wird als Konvektions-Diffusions-Fall bezeichnet, wobei für $0 < \varepsilon << 1$ der Konvektionsanteil dominiert.

Im ersten Teil dieses Kapitels werden wir uns mit der Frage nach der Existenz einer Lösung von (1.1) beschäftigen. Es sollen einige grundlegende Begriffe eingeführt und wesentliche Resultate zusammengetragen werden. Im zweiten Teil befassen wir uns mit der Methode der Finiten Differenzen, welche als Grundlage zur numerischen Lösung von (1.1) dienen wird. Im dritten Teil wird eine stabile Diskretisierung für das singulär gestörte Randwertproblem entwickelt und im vierten Teil werden diese Resultate auf den zweidimensionalen Fall übertragen.

Bei den nun folgenden Ausführungen orientieren wir uns im Wesentlichen an den Werken von [RST96] und [Ha86].

1.1 Klassische und schwache Lösungen

Im Allgemeinen kann man für das Randwertproblem (1.1) nur eine Lösung im klassischen Sinne erhalten, wenn

- b, c, f und die Daten der Randbedingungen hinreichend glatt sind,
- der Rand Γ glatt ist,
- die Randbedingungen einigen zusätzlichen Forderungen genügen.

Beispiel 1.1 *Betrachtet man das Randwertproblem* ($n = 2$)

$$\begin{aligned} -\Delta u &= 0 \qquad \text{für } x \in \Omega = (0,1)^2, \\ u &= x_1^2 \qquad \text{auf } \partial\Omega, \quad \text{mit } x = (x_1, x_2), \end{aligned}$$

dann existiert keine Lösung in $C^2(\Omega)$. *Sonst erhielte man* $u_{x_1x_1}(0,0) = 2$ *und* $u_{x_2x_2}(0,0) = 0$, *was im Widerspruch zur Differentialgleichung steht.*

Um Fuktionenräume zu erhalten, in denen glatte Lösungen im klassischen Sinne existieren, betrachten wir die sogenannten Hölderräume. Dazu definieren wir:

Definition 1.2 *Sei* $\alpha \in (0,1)$. *Eine Funktion* $\omega : \Omega \to \mathbb{R}$ *heißt* Hölder-stetig *auf* Ω *mit Exponenten* α, *wenn*

$$|\omega|_\alpha^\Omega := \sup_{x_1, x_2 \in \Omega} \frac{|\omega(x_1) - \omega(x_2)|}{dist(x_1, x_2)^\alpha} < \infty.$$

Dabei ist $dist(x_1, x_2) := (\sum_{i=1}^n (x_{1,i} - x_{2,i})^2)^{\frac{1}{2}}$. *Für* $k \in \mathbb{N}$ *definiert man damit die* Hölderräume:

$$C^{k,\alpha}(\Omega) := \left\{ \omega \in \Omega \ : \ \frac{\partial^i}{\partial x^i}\omega \text{ ist Hölder-stetig } (i = 0, \ldots, k) \right\}.$$

Existenztheorien, welche zu Lösungen $u \in C^{2,\alpha}(\Omega)$ führen, erfordern Ränder mindestens der Klasse $C^{2,\alpha}$. Aus praktischer Sicht ist diese Bedingung zu restriktiv (vgl. z.B. [ADN59]). Es ist realistischer, Lösungen in $C(\bar{\Omega}) \cap C^2(\Omega)$ zu suchen. Man sagt, Ω *ist ein Gebiet mit regulärem Rand*, wenn $\partial\Omega$ zu der Klasse

$C^{0,1}$ gehört. Das bedeutet, dass der Rand lokal durch Lipschitz-stetige Funktionen beschrieben werden kann. Im Zweidimensionalen gehört ein polygonal berandetes Gebiet ohne Lücken zu dieser Klasse.

Grob gesagt kann man für Probleme mit stetigen Daten und regulärem Rand eine klassische Lösung $u \in C(\bar{\Omega}) \cap C^2(\Omega)$ erwarten. Ein leicht abgeschwächtes Resultat stammt von [Mi77]:

Satz 1.3 *Betrachte die elliptische Differentialgleichung (1.1) mit homogenen Dirichlet-Randbedingungen. Für b, c und f Hölder-stetige Funktionen auf $\bar{\Omega}$, $c \geq 0$ und Ω Gebiet mit regulärem Rand existiert eine eindeutige Lösung $u \in C(\bar{\Omega}) \cap C^2(\Omega)$.*

Elliptische Operatoren erfüllen das *Maximums- und Vergleichsprinzip*. Folgendes Resultat stammt von [PW67]:

Satz 1.4 (Klassisches Vergleichsprinzip)
Sei $c \geq 0$ und $v, w \in C(\bar{\Omega}) \cap C^2(\Omega)$ mit

$$\begin{aligned} |(Lv)(x)| &\leq (Lw)(x) \qquad \forall x \in \Omega, \\ |v(x)| &\leq w(x) \qquad \forall x \in \partial\Omega. \end{aligned}$$

Dann gilt für alle $x \in \bar{\Omega}$: $\quad |v(x)| \leq w(x)$.

Die Lösung eines gegebenen Randwertproblems wird häufig durch dieses Prinzip mittels einer geeigneten Schrankenfunktion w abgeschätzt.

In Gebieten mit Ecken, welche eine wichtige Rolle im Bereich der Anwendung spielen, ist die Theorie der klassischen Lösung häufig nicht ausreichend. Eine alternative Möglichkeit stellt die Verwendung von *schwachen Lösungen* dar. Dazu benötigen wir vorab einige Hilfsmittel.

Definition 1.5 *Sei $u \in L_2(\Omega)$ und $\alpha \in \mathbb{N}^{\mathbb{N}}$ ein Multiindex der Ordnung $|\alpha| > 0$. Weiter existiere ein $u_\alpha \in L_2(\Omega)$, so dass*

$$\forall \varphi \in C_0^\infty(\Omega) \quad \int_\Omega u \partial^\alpha \varphi \, dx = (-1)^{|\alpha|} \int_\Omega u_\alpha \varphi \, dx \tag{1.2}$$

gilt. Wir bezeichnen dann $\partial^\alpha u := u_\alpha$ als α-te schwache Ableitung von u und sagen $\partial^\alpha u$ existiert schwach in $L_2(\Omega)$.

Bemerkung 1.6 *∂^α ist wohldefiniert. Gilt (1.2) für u_α und $v_\alpha \in L_2(\Omega)$, dann ist $\int_\Omega (v_\alpha - u_\alpha)\varphi\, dx = 0$ für alle $\varphi \in C_0^\infty(\Omega)$ und demnach gilt $v_\alpha = u_\alpha$ f.ü. in Ω.*

Definition 1.7 *Sei $m \in \mathbb{N}$. Mit $H^m(\Omega)$ bezeichnen wir den Raum, dessen Elemente in $L_2(\Omega)$ liegen und deren schwache Ableitungen bis zur Ordnung m ebenfalls in $L_2(\Omega)$ sind:*

$$H^m(\Omega) := \{u \in L_2(\Omega) \,:\, \partial^\alpha u \in L_2(\Omega) \text{ für } |\alpha| \le m\}\,. \tag{1.3}$$

Für $u, v \in H^m(\Omega)$ führen wir

$$< u, v >_\Omega := < u, v >_{0,\Omega} := \int_\Omega uv\, dx \quad , \quad ||u||_\Omega := (< u, u >_{0,\Omega})^{\frac{1}{2}}, \tag{1.4}$$

$$< u, v >_{m,\Omega} := \sum_{0 \le |\alpha| \le m} < \partial^\alpha u, \partial^\alpha v >_\Omega \quad , \quad ||u||_{m,\Omega} := (< u, u >_{m,\Omega})^{\frac{1}{2}} \tag{1.5}$$

ein.

(1.3) wird auch als Sobolevraum bezeichnet. Im weiteren Verlauf wird $< ., . >_{m,\Omega}$ bzw. $||.||_{m,\Omega}$ der Einfachheit halber als $< ., . >$ bzw. $||.||$ bezeichnet werden. Eine Unterscheidung wird nur in den Fällen vorgenommen, bei denen keine Eindeutigkeit vorliegt.

Lemma 1.8 *Die Abbildung*

$$< ., . >_{m,\Omega} :\; H^m(\Omega) \times H^m(\Omega) \to \mathbb{C}, \quad (u, v) \mapsto < u, v >_{m,\Omega}$$

ist ein Skalarprodukt auf $H^m(\Omega)$ und mit diesem Skalarprodukt wird $H^m(\Omega)$ zum Hilbertraum.

Beweis: s. [Ag65].

Kommen wir nun zurück zur schwachen Lösung einer Differentialgleichung. Wir betrachten einen gegebenen Hilbertraum V mit einer Norm $||.||_V$ und einem Skalarprodukt $< ., . >$. Bei der Diskretisierung von Differentialgleichungen 2. Ordnung auf einem Gebiet Ω ist V für gewöhnlich ein Teilraum des Sobolevraumes $H^1(\Omega)$. Auf diesem formt man die Differentialgleichung in ein äquivalentes Variationsproblem um:
Finde $u \in V$, so dass

$$a(u, v) = f(v) \qquad \forall v \in V \tag{1.6}$$

gilt. Dabei ist $a(.,.)$ eine gegebene Bilinearform auf $V \times V$ und $f(.)$ ein gegebenes stetiges lineares Funktional auf V_0. Ein u, welches (1.6) erfüllt, bezeichnet man als schwache Lösung von $Lu = f$.

Betrachten wir nun ein elliptisches Randwertproblem und formen es in seine schwache Formulierung um. Dazu sei der Rand Γ in drei stückweise disjunkte Teile Γ_1, Γ_2 und Γ_3 zerlegt. Wir betrachten das Randwertproblem

$$\left.\begin{array}{rcll} -\varepsilon\Delta u + b(x)\cdot\nabla u + c(x)u &=& f(x) & \text{in } \Omega, \\ u &=& 0 & \text{auf } \Gamma_1, \\ \varepsilon\frac{\partial u}{\partial n} &=& 0 & \text{auf } \Gamma_2, \\ \varepsilon\frac{\partial u}{\partial n} + \beta u &=& g & \text{auf } \Gamma_3. \end{array}\right\} \tag{1.7}$$

Wir definieren

$$V = \{v \in H^1(\Omega) \ : \ v = 0 \text{ auf } \Gamma_1\}$$

und führen die Bilinearform

$$a(v,w) := \varepsilon < \nabla v, \nabla w > + < b\cdot\nabla v + cv, w > + \int_{\Gamma_3} \beta v w \tag{1.8}$$

sowie die Linearform

$$f(v) :=< f, v > + \int_{\Gamma_3} g v \tag{1.9}$$

ein. Damit lässt sich nun (1.7) umformulieren:
Finde $u \in V$, so dass

$$a(u,v) = f(v) \qquad \forall v \in V. \tag{1.10}$$

Das Lemma von Lax-Milgram liefert hinreichende Bedingungen für die Existenz und Eindeutigkeit der Lösung von (1.6):

Lemma 1.9 (Lax-Milgram, 1954)
Sei $a(.,.)$ eine stetige Bilinearform auf $V \times V$, d.h.

$$\exists \beta > 0 \ \forall v, w \in V \ : \ |a(v,w)| \le \beta ||v|| \cdot ||w||.$$

Weiter sei $a(.,.)$ V-elliptisch (oder koerziv), d.h.

$$\exists \alpha > 0 \ \forall v \in V \ : \ a(v,v) \ge \alpha ||v||^2.$$

Dann existiert für (1.6) eine eindeutige Lösung.

Um für (1.10) Existenz und Eindeutigkeit einer schwachen Lösung zu garantieren, benötigen wir also nach Lax-Milgram die Stetigkeit und die V-Elliptizität

der Bilinearform $a(.,.)$. Standardmäßig erhält man die V-Elliptizität, indem man

$$c - \frac{1}{2} \div b \geq \omega > 0, \quad \beta > 0, \quad b \cdot n \geq 0 \text{ auf } \Gamma_2 \cup \Gamma_3 \text{ und } \quad \Gamma_1 \neq \emptyset$$

fordert. Um glatte Lösungen zu erhalten (z.B. $u \in H^2(\Omega)$), benötigt man zusätzliche Voraussetzungen an die Daten, die Struktur des Randes und die Randwertbedingungen. Für homogene Randwertbedingungen in einem Gebiet $\Omega \subseteq \mathbb{R}^2$ benötigt man im Allgemeinen die Konvexität von Ω, um eine Lösung in $H^2(\Omega)$ zu erhalten.

Das Maximums- und Vergleichsprinzip existiert auch für schwache Lösungen.

Satz 1.10 (Schwaches Maximumsprinzip)
Sei $a(.,.)$ eine H_0^1-elliptische Bilinearform auf $H^1 \times H^1$. Dann gilt für alle $v \in H^1(\Omega)$, die die Bedingungen

$$\begin{aligned} v &\leq 0 \qquad \text{auf } \partial\Omega, \\ a(v,w) &\leq 0 \qquad \forall w \in H^1(\Omega) \ : \ w \geq 0 \end{aligned}$$

erfüllen, dass auch $v \leq 0$ in $\bar{\Omega}$ gelten muss.

Beweis: s. [GT83]

1.2 Methode der Finiten Differenzen

In den wenigsten Fällen ist es möglich, eine partielle Differentialgleichung exakt zu lösen. Man bedient sich daher Verfahren aus der Numerik, um an einigen diskreten Punkten eine Näherungslösung zu bestimmen. Ein Verfahren dazu ist die Methode der Finiten Differenzen. Wir betrachten hierfür zunächst den eindimensionalen Fall eines nicht singulär gestörten Problems und übertragen diesen dann auf singulär gestörte Probleme und den zweidimensionalen Fall.

Es sei $\Omega = [0,1]$ gegeben. Hierauf betrachten wir

$$Lu := -u'' + b(x)u' + c(x)u = f(x), \quad u(0) = u(1) = 0 \tag{1.11}$$

mit b, c, f hinreichend glatt und $c(x) \geq 0$ gegeben. Die Methode der Finiten Differenzen ist dadurch charakterisiert, dass man die Ableitungen durch

Differenzenquotienten ersetzt. Wir definieren zunächst ein Teilgebiet $\Omega_h \subseteq \Omega$, welches aus isolierten Punkten besteht:

$$\Omega_h := \{(x_0, \ldots, x_N) \in \Omega : x_i/h \in \mathbb{Z}, i = 0, \ldots, N\}.$$

Man bezeichnet Ω_h auch als Gitter (über dem Gebiet Ω). Analog definiert man für den Rand von Ω

$$\partial\Omega_h := \{(x_0, \ldots, x_N) \in \partial\Omega : x_i/h \in \mathbb{Z}, i = 0, \ldots, N\}.$$

Unter Verwendung der Gitterpunkte x_i wird nun die Differentialgleichung diskretisiert, wobei die Unbekannten $u_i (i = 0, \ldots, N)$ eine Näherung der $u(x_i)$ darstellen. So benutzt man zentrale Differenzen, um $u'(x)$ auf natürliche Weise zu approximieren:

$$(D^0 u)(x) := [u(x + h) - u(x - h)]/(2h).$$

Durch Verkettung der Vorwärts- und Rückwärtsdifferenzen

$$(D^+ u)(x) := [u(x + h) - u(x)]/h \text{ und}$$
$$(D^- u)(x) := [u(x) - u(x - h)]/h$$

erhält man die folgende zentrale Approximation für $u''(x)$:

$$(D^+ D^- u)(x) = [u(x - h) - 2u(x) + u(x + h)]/h^2.$$

Definition 1.11 *Es sei $(Du)(x)$ eine Finite-Differenzen-Approximation von $\partial^\alpha u(x)$ $(\alpha \in \mathbb{N})$. Gilt*

$$|(Du)(x) - \partial^\alpha u(x)| = O(h^p) \quad (p \in \mathbb{N}),$$

so sagt man D approximiert $\partial^\alpha u$ von der Ordnung p.

Die Ordnung einer Finiten-Differenzen-Approximation hängt von der Glattheit von u ab. Über Taylorentwicklung erhält man z.B.

$$u(x \pm h) = u(x) \pm hu'(x) + \frac{h^2}{2}u''(x) \pm \frac{h^3}{6}u'''(x) + R_4$$

mit

$$R_4 = \int_x^{x \pm h} [u'''(\xi) - u'''(x)] \frac{(x \pm h - \xi)^2}{2} d\xi.$$

Also gilt

$$\left|(D^+D^-u)(x) - u''(x)\right| \leq K \cdot h^2,$$

falls $u \in C^4$ (es ist möglich, diese Bedingung auf $u \in C^{3,1}$ abzuschwächen). In diesem Fall ist D^+D^- von zweiter Ordnung. Hat man hingegen nur $u \in C^3$, so sinkt die Ordnung auf eins.

Unter Verwendung der Finiten-Differenzen-Approximationen erhält man nun punktweise für (1.11)

$$\begin{aligned} -D^+D^-u_i + b_iD^0u_i + c_iu_i &= f_i, \quad (i = 1, \ldots, N-1) \\ u_0 = u_N &= 0. \end{aligned} \tag{1.12}$$

Hierbei bezeichnen b_i, c_i und f_i den Wert der entsprechenden Funktion an der Stelle x_i. Daraus ergibt sich ein tridiagonales lineares Gleichungssystem für $i = 1, \ldots, N-1$ der Form

$$(-\frac{1}{h^2} - \frac{1}{2h}b_i)u_{i-1} + (c_i + \frac{2}{h^2})u_i + (-\frac{1}{h^2} + \frac{1}{2h}b_i)u_{i+1} = f_i, \tag{1.13}$$

mit $u_0 = u_N = 0$.

Es stellt sich nun die Frage, welche Eigenschaften die Finite Differenzen Approximation besitzt und was sich über die Differenz $|u(x_i) - u_i|$ sagen lässt.

Man schreibt das Differenzenschema um in

$$L_hu_h = f_h, \quad \text{mit}$$

$$\begin{aligned} u_h &:= (u_h(x_0), u_h(x_1), \ldots, u_h(x_N))^T \\ &:= (u_0, u_1, \ldots, u_N)^T \text{ und} \\ f_h &:= (f(x_0), f(x_1), \ldots, f(x_N))^T. \end{aligned}$$

Funktionen, welche auf dem Gitter Ω_h definiert werden, bezeichnet man auch als Gitterfunktionen. Die Restriktion einer Funktion $w \in C[0,1]$ auf das Gitter wird mittels des Operators R_h gegeben, d.h.

$$R_hw := (w(x_0), w(x_1), \ldots, w(x_N))^T.$$

Damit lässt sich nun das Differenzenschema in der Form $R_h(Lu) = L_hu_h$ schreiben. Auf dem Raum der Gitterfunktionen führen wir die diskrete Maximumsnorm

$$||w_h||_{\infty,d} := \max_i |w_h(x_i)|$$

ein. An Stellen, wo die Bezeichung klar ist, verwenden wir der Einfachheit halber $||w_h||_\infty$ anstelle von $||w_h||_{\infty,d}$.

Definition 1.12 *Ein Differenzenschema $L_h u_h$ heißt konsistent von der Ordnung k bzgl. der diskreten Maximumsnorm, wenn*

$$||L_h R_h u - R_h L u||_\infty \leq K h^k,$$

mit K und k positive Konstanten unabhänig von h.

Wendet man den diskreten Operator L_h auf den Fehler an den inneren Gitterpunkten an, so erhält man

$$\begin{aligned} L_h(R_h u - u_h) &= L_h R_h u - f_h \\ &= L_h R_h u - R_h L u. \end{aligned} \tag{1.14}$$

Um den Fehler $R_h u - u_h$ aus (1.14) zu bestimmen, ebenso wie die Konsistenzordnung, führt man den Begriff der Stabilität ein.

Definition 1.13 *Ein diskretes Problem $L_h u_h = f_h$ heißt stabil in der diskreten Maximumsnorm, wenn eine Konstante $K > 0$ unabhängig von h existiert, so dass*

$$||u_h||_\infty \leq K ||L_h u_h||_\infty$$

für alle Gitterfunktionen u_h gilt.

Zuletzt führen wir den Begriff der Konvergenz ein.

Definition 1.14 *Ein Differenzenverfahren heißt konvergent von der Ordnung k in der diskreten Maximumsnorm, wenn positive Konstanten K und k unabhängig von h existieren, so dass*

$$||u_h - R_h u||_\infty \leq K h^k$$

gilt.

Damit erhält man das zentrale Resultat der klassischen Konvergenztheorie für Finite Differenzen Verfahren:

$$\text{Stabilität} + \text{Konsistenz} \Longrightarrow \text{Konvergenz}.$$

Die Konsistenzordnung eines Finiten-Differenzen-Schemas lässt sich meistens über Taylorentwicklung relativ einfach bestimmen. Das Bestimmen der Stabilität ist i.A. nicht so leicht. Hierzu verwendet man standardmäßig die Theorie der *M-Matrizen*.

Definition 1.15 *Eine Matrix $A = (a_{ij})$, $i, j \in I$ heißt M-Matrix, wenn*

$$a_{ii} > 0 \quad \forall\, i \in I, \quad a_{ij} \leq 0 \quad \forall\, i \neq j, \tag{1.15}$$

$$A \text{ nichtsingulär und } A^{-1} \geq 0. \tag{1.16}$$

Dabei bedeutet die Notation $A \geq 0$, dass $a_{ij} \geq 0 \quad \forall\, i, j \in I$ gilt. Analog notiert man für Vektoren $v \geq 0$, falls $v_j \geq 0 \quad \forall\, j \in I$ gilt. Die Bedingung (1.15) lässt sich meistens recht leicht überprüfen, wohin gegen für (1.16) weitere Hilfsmittel benötigt werden.

Definition 1.16 *(i) Ein Index $i \in I$ heißt direkt mit j verbunden, falls $a_{ij} \neq 0$. Ein Index $i \in I$ heißt mit j verbunden, falls eine Verbindung*

$$i = \alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_k = j \text{ mit } a_{\alpha_{l-1}\alpha_l} \neq 0 \quad (1 \leq l \leq k)$$

existiert.

(ii) A heißt irreduzibel, falls jedes $i \in I$ mit jedem $j \in I$ verbunden ist.

Für den Fall $A = L_h$ sind zwei Indizes $x, y \in \Omega_h$ genau dann direkt verbunden, wenn $x = y$ gilt oder wenn x ein Nachbarpunkt von y ist. Beliebige Punkte $x, y \in \Omega_h$ lassen sich offensichtlich durch eine Kette von benachbarten Punkten verbinden. L_h ist also irreduzibel.

Definition 1.17 *(i) A heißt diagonaldominant, wenn*

$$\sum_{\substack{i \neq j \\ i \in I}} |a_{ij}| < |a_{jj}| \quad \forall j \in I.$$

(ii) A heißt irreduzibel diagonaldominant, wenn A irreduzibel,

$$\forall j \in I \quad : \quad \sum_{\substack{i \neq j \\ i \in I}} |a_{ij}| \leq |a_{jj}| \text{ und}$$

$$\exists j \in I \quad : \quad \sum_{\substack{i \neq j \\ i \in I}} |a_{ij}| < |a_{jj}|.$$

Man beachte, dass eine irreduzible und diagonaldominante Matrix zwar irreduzibel diagonaldominant ist, die Umkehrung jedoch nicht zu gelten braucht.

Man definiert nun $D := diag(A)$ und zerlegt damit die Matrix A in $A = B - D$. Damit erhält man

Lemma 1.18 *A erfülle (1.15). Dann ist A genau dann eine M-Matrix, wenn der Spektralradius $\rho(D^{-1}B) < 1$ ist. Dabei ist Diagonaldominanz oder die irreduzible Diagonaldominanz von A ein hinreichendes Kriterium für $\rho(D^{-1}B) < 1$.*

Beweis: s. [Ha86].

Es lässt sich also festhalten, dass eine Matrix A, die (1.15) erfüllt und zusätzlich diagonaldominant oder irreduzibel diagonaldominant ist, eine M-Matrix ist.

Satz 1.19 *Eine irreduzible M-Matrix A hat eine positive Inverse: $A^{-1} > 0$.*

Beweis: s. [Ha86].

Mittels des folgenden Satzes lässt sich dann die Stabilität eines Finiten-Differenzen-Schemas zeigen.

Satz 1.20 *Sei A eine Matrix mit $a_{ij} \leq 0 \quad \forall\, i \neq j$. Dann ist A genau dann eine M-Matrix, wenn ein Vektor $e > 0$ existiert, so dass $Ae > 0$ gilt. Weiterhin gilt dann*

$$||A^{-1}||_\infty \leq \frac{||e||_\infty}{\min_k (Ae)_k}.$$

Beweis: s. [Bo81].

Für den Fall, dass L_h eine M-Matrix ist, kann man also die Stabilität des Schemas zeigen:

$$\begin{aligned} ||u_h||_\infty &= ||L_h^{-1} L_h u_h||_\infty \leq ||L_h^{-1}||_\infty \cdot ||L_h u_h||_\infty \\ &\leq \underbrace{\frac{||e||_\infty}{\min_k (L_h e)_k}}_{:=K} ||L_h u_h||_\infty. \end{aligned}$$

Als Beispiel für eine stabile und konsistente Diskretisierung betrachten wir das Problem der Poisson-Gleichung

$$\begin{aligned} Lu := -u'' &= f \quad \text{in } \Omega, \\ u &= \phi \quad \text{auf } \Gamma. \end{aligned} \tag{1.17}$$

Mittels zentraler Differenzen erhält man den folgenden diskreten Differentialoperator

$$L_h u = \frac{1}{h^2} \left[\begin{array}{ccc} -1, & 2, & -1 \end{array} \right] u.$$

Offensichtlich ist (1.15) erfüllt und wie schon oben erwähnt, ist L_h irreduzibel diagonaldominant. Folglich ist L_h eine M-Matrix und demnach die Diskretisierung stabil. Über Taylorentwicklung zeigt man leicht die Konsistenz der Diskretisierung, woraus dann insgesamt die Konvergenz des Verfahrens folgt.

1.3 Upwind-Verfahren

In diesem Abschnitt beschäftigen wir uns mit Differenzenverfahren für das singulär gestörte Randwertproblem

$$Lu := -\varepsilon u'' + b(x)u' + c(x)u = f(x) \text{ auf } (0,1), \quad u(0) = u(1) = 0. \tag{1.18}$$

Es seien hierfür $b(x) \neq 0$, $c(x) \geq 0$ für alle $x \in [0,1]$ und b, c und f hinreichend glatt. Für $b > 0$ hat man einen Randlayer bei $x = 1$ und für $b < 0$ einen Randlayer bei $x = 0$. Es sei bemerkt, dass die Fälle $b > 0$ und $b < 0$ über die Transformation $x \mapsto 1 - x$ äquivalent zueinander sind.

Wir wollen die Eigenschaften der Finiten-Differenzen-Diskretisierung an einem einfachen Beispiel untersuchen. Dazu betrachten wir das Randwertproblem

$$-\varepsilon u'' - u' = 0, \quad \text{in} I = (0,1) \tag{1.19}$$

$$u(0) = 0 \quad , \quad u(1) = 1. \tag{1.20}$$

Die exakte Lösung hierfür ist

$$u(x) = \frac{1 - \exp(-x/\varepsilon)}{1 - \exp(-1/\varepsilon)}. \tag{1.21}$$

Diskretisiert man wie üblich u' und u'' mit zentralen Differenzen, so erhält man für die Differentialgleichung (1.19) den 3-Punkt-Stern

$$-\varepsilon u'' - u' \simeq \left[\; -\frac{\varepsilon}{h^2} + \frac{1}{2h}, \quad \frac{2\varepsilon}{h^2}, \quad -\frac{\varepsilon}{h^2} - \frac{1}{2h} \; \right] u. \tag{1.22}$$

Die Lösung des hierbei entstehenden linearen Gleichungssystems lässt sich leicht direkt berechnen. Dazu wählen wir den Ansatz

$$u_h^i = \alpha \cdot 1 + \beta \cdot r^i, \quad (i = 0, \ldots, N).$$

Man beachte, daß 1 eine Lösung von (1.19) ist. Setzt man den Ansatz in die diskretisierte Differentialgleichung (1.22) ein, so erhält man

$$\begin{aligned}
& (-\frac{\varepsilon}{h^2}+\frac{1}{2h})r^{i-1}+\frac{2\varepsilon}{h^2}r^i-(\frac{\varepsilon}{h^2}+\frac{1}{2h})r^{i+1}=0 \\
\Longrightarrow\quad & (-\frac{\varepsilon}{h^2}+\frac{1}{2h})+\frac{2\varepsilon}{h^2}r-(\frac{\varepsilon}{h^2}+\frac{1}{2h})r^2=0 \\
\Longrightarrow\quad & r_{1,2}=\frac{1}{\varepsilon+\frac{h}{2}}(\varepsilon\pm\frac{h}{2}) \\
\Longrightarrow\quad & r_1=1,\quad r_2=\frac{2\varepsilon-h}{2\varepsilon+h}:=r
\end{aligned}$$

Aus den Randbedingungen (1.20) ergibt sich dann

$$\alpha=\frac{1}{1-r^N},\quad \beta=-\frac{1}{1-r^N}.$$

Für $\varepsilon << h$ ist $r\approx -1$ also $1-r^N\approx 0,\ (N\equiv 0 \bmod 2)$. Daraus ergibt sich für die Lösung

$$u_h^i=\frac{1-r^i}{1-r^N}\approx \underbrace{\frac{1}{1-r^N}}_{\to\infty}\underbrace{(1-(-1)^i)}_{\text{oszilliert}},\quad (i=1\ldots N).$$

Die gewählte Diskretisierung mit zentralen Differenzen für den Konvektionsanteil ist also nicht stabil.

Kommen wir zu dem allgemeineren Fall (1.18) zurück. Mittels zentraler Differenzen erhält man

$$L_h=\left[\ -\frac{\varepsilon}{h^2}-\frac{1}{2h}b_i,\quad c_i+\frac{2\varepsilon}{h^2},\quad -\frac{\varepsilon}{h^2}+\frac{1}{2h}b_i\ \right].$$

Dies ergibt eine M-Matrix und folglich Stabilität nur unter der Annahme, dass

$$h\le h_0(\varepsilon)=\frac{2\varepsilon}{\max|b|}$$

gilt. Für den Fall $\varepsilon << 1$ benötigt man also unvertretbar kleine Schrittweiten h.

Ein alternativer, heuristischer Erklärungsanstz für das Scheitern von zentralen Differenzen liefert eine Betrachtung der Randwerte des Beispiels (1.19). Für den Fall $\varepsilon << h$ erhält man hierbei im Wesentlichen $D^0u_i=0$. Das bedeutet, dass $u_{N-2}\cong u_N=1$ gilt und somit u_{N-2} eine schlechte Approximation an $u(x_{N-2})$ darstellt.

Die Idee besteht also darin, bei der Approximation von $u'(x_{N-1})$ die Verwendung des Punktes u_{N-2} zu vermeiden. Das einfachste Schema, das diese Forderung erfüllt, ist

$$u'(x_i) \simeq \frac{u_{i+1} - u_i}{h}. \tag{1.23}$$

Man erhält hiermit für (1.19)

$$-\varepsilon u'' - u' \simeq \left[-\frac{\varepsilon}{h^2}, \quad \frac{2\varepsilon}{h^2} + \frac{1}{h}, \quad -\frac{\varepsilon}{h^2} - \frac{1}{2h} \right] u.$$

Dies ist offensichtlich eine M-Matrix. Mit dem selben Ansatz wie oben ergibt sich dann

$$\begin{aligned} & r_1 = 1, \quad r_2 = \frac{\varepsilon}{\varepsilon + h} := r \\ \Longrightarrow\ & u_h^i = \frac{1 - r^i}{1 - r^N}. \end{aligned}$$

Für $\varepsilon << h$ ist $r \approx 0$ und es gilt $0 \leq u_h^i \leq 1$. Die Diskretisierung ist also stabil.

Verallgemeinert man dies auf den Fall, in dem b entweder positiv oder negativ ist, so erhält man für (1.18) das Schema

$$Lu \simeq \frac{1}{h^2} \left[-\varepsilon + hb^+, \quad 2\varepsilon + h|b| + h^2 c, \quad -\varepsilon + hb^- \right] u, \tag{1.24}$$

mit $b^+ := \max\{0, b\},\ b^- := \min\{b, 0\}$.

Dieses Schema wird auch als *(einfaches) Upwind-Verfahren* bezeichnet. *Upwind* bedeutet hierbei, dass die Finite-Differenzen-Approximation des Konvektionsterms in Upstream-Richtung des jeweiligen Gitterpunktes gebildet wird. Man sieht leicht, dass dieses Schema eine M-Matrix liefert. Folglich gilt

Lemma 1.21 *Die Koeffizienten-Matrix für das Upwind-Verfahren ist eine M-Matrix. Unter der Annahme, dass $b(x) \neq 0$ für alle $x \in [0, 1]$ gilt, ist das Upwind-Verfahren gleichmäßig stabil (abhängig von ε); d.h.*

$$||u_h||_\infty \leq C||L_h u_h||_\infty,$$

mit einer Stabilitätskonstanten C unabhängig von ε und h.

Beweis: s. [RST96].

Um die Stabilität des Upwind-Verfahrens zu gewährleisten, zahlt man einen Preis in der Genauigkeit. Während die Diskretisierung mittels zentraler Differenzen von 2. Ordnung ist, hat man hierbei nur die Ordnung 1.

Satz 1.22 *Sei $b \geq b_0 > 0$ und $c \geq 0$. Dann existiert eine positive Konstante b_0^*, welche nur von b_0 abhängt, so dass für den Fehler des Upwind-Verfahrens an den inneren Punkten $\{x_i \,:\, 1, \ldots, N-1\}$*

$$|u(x_i) - u_i| \leq \begin{cases} Ch(1 + \varepsilon^{-1} \exp(-b_0^*(1-x_i)/\varepsilon)) & \text{für } h \leq \varepsilon \\ Ch + C\exp(-b_0^*(1-x_{i+1})/\varepsilon) & \text{für } h \geq \varepsilon \end{cases}$$

gilt.

Beweis: Siehe [RST96].

Satz (1.22) zeigt, dass man außerhalb der Layers (z.B. auf $[0, 1-\delta]$ für festes $\delta > 0$) Konvergenz der Ordnung 1 mit einer Konstanten unabhängig von ε erhält. Eine Reihe von Problemen sind immer noch ungelöst, was die Konstruktion eines Upwind-Schemas angeht, welches außerhalb des Layers hohe Konvergenzordnung erzielt.

Eine mögliche Vorgehensweise besteht in der Umformung von (1.24). Man erhält

$$-(\varepsilon + b_i \frac{h}{2}) D^+ D^- u_i + b_i D^0 u_i + c_i u_i = f_i, \quad u_0 = u_N = 0. \tag{1.25}$$

Dies ähnelt dem zentralen Differenzen-Schema, aber der Diffusionskoeffizient ist hierbei $\varepsilon + b_i \frac{h}{2}$ anstelle von ε. Das bedeutet, dass das einfache Upwind-Verfahren äquivalent zu einem zentralen Differenzen-Schema ist, welches auf eine modifizierte Differentialgleichung angewandt worden ist. Für $\varepsilon > b_i h/2$ ist der dominante Konvektionsteil in $O(\varepsilon)$, aber für den interessanteren Fall $\varepsilon < b_i h/2$ in $O(b_i h/2)$. Man spricht bei (1.25) von *künstlicher Viskosität* bzw. *künstlicher Diffusion*. Dies verdeutlicht die Strategie einer einfachen Idee: Addiere künstliche Viskosität zu einer gegebenen Differentialgleichung, um diese zu stabilisieren.

Es zeigt sich, dass zu viel künstliche Viskosität die Lösung verschmiert, d.h. der berechnete Layer wird zu breit. Im Zweidimensionalen verstärkt sich dieses Problem noch.

Allgemein lässt sich die Hinzunahme von künstlicher Viskosität durch einen Parameter σ ausdrücken, d.h.

$$-\varepsilon\sigma(q(x_i)) D^+ D^- u_i + b_i D^0 u_i + c_i u_i = f_i, \quad u_0 = u_N = 0 \tag{1.26}$$

mit

$$q(x) = \frac{b(x)h}{2\varepsilon}.$$

Für $\sigma(q) = 1+q$ erhält man das einfache Upwind-Verfahren. Weitere bekannte Verfahren sind

$$\sigma(q) = \max(1, q), \qquad \sigma(q) = \sqrt{1+q^2}$$
$$\sigma(q) = 1 + q^2/(1+q) \quad \text{(Samarskij's Upwind Schema)}.$$

Nimmt man an, dass $b(x) \geq b_0 > 0,\ c \geq 0,\ \sigma(q) > q$ und $|\sigma(q) - 1| \leq \min(q, Mq^2)$ gilt, so kann man zeigen, dass bei diesen Beispielen für festes ε Konvergenz der Ordnung 2 vorliegt. Allgemein lässt sich aber auch hierfür nur Konvergenz 1. Ordnung gleichmäßig in ε zeigen (vgl. [KT78]).

Ein Schema, welches auch außerhalb des Layers 2. Ordnung erzielt, wurde 1977 von Stoyan eingeführt:

$$-\varepsilon\sigma(q(x_{i-\alpha}))D^+D^-u_i + b(x_{i-\alpha})D^0u_i + c(x_{i-\alpha})u_i = f(x_{i-\alpha})$$

mit der geschifteten Gleichung

$$x_{i-\alpha} := (i-\alpha)h \text{ und } \alpha = \alpha(q) := (\sigma(q)-1)/(2q),$$

wobei q die nichtlineare Gleichung

$$q = \frac{h}{2\varepsilon} b(x_i - \alpha(q)h)$$

erfüllt.

Den bisher vorgestellten Ansätzen ist gemein, dass es sich bei ihnen um Dreipunktschemata handelt. Es gibt auch Ansätze, ein Upwind-Verfahren hoher Ordnung zu entwickeln, welches mehr als drei Punkte verwendet (vgl. z.B. [GS74], [Le79] und [Gu88]).

1.4 Der zweidimensionale Fall

Wir erweitern im folgenden Kapitel die Methode der Finiten Differenzen auf die zweite Dimension. Dabei werden wir, um mit beliebigen Gebieten umgehen zu können, auch Schemata für nicht-äquidistante Schrittweiten von h betrachten. Ebenso wird das einfache Upwind-Verfahren auf den zweidimensionalen Fall erweitert. Wir betrachten dazu das Konvektions-Diffusions-Problem

$$\left.\begin{aligned} Lu := -\varepsilon\Delta u + b(x)\cdot\nabla u + c(x)u &= f(x) \quad \text{auf } \Omega := [0,1]^2, \\ u &= 0 \quad \text{auf } \Gamma := \partial\Omega \end{aligned}\right\} \tag{1.27}$$

mit $b = (b_1, b_2)$ und $0 < \varepsilon << 1$. Analog wie im eindimensionalen Fall lassen sich auch hier die Ableitungen durch Finite-Differenzen-Schemata approximieren. So ersetzt man den Laplace-Operator Δ durch die zentrale Differenz in $x-$ bzw. $y-$Richtung:

$$\begin{aligned}(\Delta u)(x,y) &\simeq (D_x^+ D_x^- + D_y^+ D_y^-)u(x,y) \\ &= \frac{1}{h^2}\begin{bmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{bmatrix} u(x,y). \end{aligned} \tag{1.28}$$

Da hierbei die Funktion $u(x,y)$ an fünf Punkten ausgewertet wird, bezeichnet man diese Diskretisierung auch als *Fünfpunktformel*. Mittels Taylorentwicklung sieht man leicht, dass diese Approximation von der Ordnung 2 ist.

Ebenso erhält man für die erste Ableitung über zentrale Differenzen

$$\begin{aligned}(\nabla u)(x,y) &\simeq (D_x^0 + D_y^0)u(x,y) \\ &= \frac{1}{2h}\begin{bmatrix} & 1 & \\ -1 & 0 & 1 \\ & -1 & \end{bmatrix} u(x,y). \end{aligned} \tag{1.29}$$

Diskretisiert man (1.27) mit (1.28) und (1.29), so erhält man nur eine M-Matrix, wenn man die Schrittweite h nach oben durch $K \cdot \varepsilon$ beschränkt. Dies führt zu unvertretbar kleinen Schrittweiten und demnach zu viel zu großen Gleichungssystemen. Ein Ausweg wurde bereits in Kap. 1.3 vorgestellt: Man muss den Konvektionsterm durch geeignete einseitige Differenzen approximieren:

$$L_h = h^{-2}\begin{bmatrix} & -\varepsilon + hb_2^- & \\ -\varepsilon - hb_1^+ & 4\varepsilon + h|b_1| + h|b_2| + h^2 c & -\varepsilon + hb_1^- \\ & -\varepsilon - hb_2^+ & \end{bmatrix}, \tag{1.30}$$

wobei $b_i^+ := \max\{0, b_i\}$, $b_i^- := \min\{0, b_i\}$ ist. Die Anwendung dieses Differenzenoperators auf eine glatte Funktion liefert über Taylorentwicklung

$$L_h u = Lu - (|b_1| u_{xx} + |b_2| u_{yy})\, h + O(h^2).$$

Den $O(h)$-Term $|b_1| u_{xx} + |b_2| u_{yy}$ bezeichnet man als *numerische Viskosität*. Während dieser Fall im Eindimensionalen identisch mit der *künstlichen Viskosität* ist, besteht im Zweidimensionalen ein Unterschied. Der Ansatz der künstlichen Viskosität besteht darin, dass man den Parameter ε bei der Diskretisierung duch $\varepsilon_h \geq \varepsilon$ ersetzt. Wählt man

$$\varepsilon_h := \max\{\varepsilon,\ h|b_1|/2,\ h|b_2|/2\} \text{ oder } \varepsilon_h := \varepsilon + \frac{h}{2}\max|b_i|,$$

so führt das zentrale Differenzenverfahren

$$L_h = \varepsilon_h h^{-2} \begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix} + \frac{1}{2} h^{-1} \begin{bmatrix} & b_2 & \\ -b_1 & 0 & b_1 \\ & -b_2 & \end{bmatrix} + c \tag{1.31}$$

zu einer M-Matrix. Hierbei wird der Konvektionsterm zwar mit einer Konsistenzordnung von zwei diskretisiert, jedoch beträgt der Fehler des Diffusionsterms $O(\varepsilon_h - \varepsilon)$. Dieser Fehler stimmt für den praktisch relevanten Fall $h > 2\varepsilon/||b||_\infty$ mit $O(h)$ überein. Über Taylorentwicklung erhält man auch hierfür

$$L_h u = Lu - (\varepsilon_h - \varepsilon)\Delta u + h + O(h^2).$$

Der Differenzbetrag $-(\varepsilon_h - \varepsilon)$ wird als künstliche Viskosität bezeichnet.

Nachdem auch für den zweidimensionalen Fall eine stabile Diskretisierung entwickelt worden ist, werden diese Ergebnisse nun auf beliebige Gitter, d.h. mit variabler Schrittweite h übertragen. Wir betrachten dazu einen Punkt $(x, y) \in \Omega$ und seine Nachbarpunkte $(x - h_1^x, y), (x + h_2^x, y), (x, y - h_1^y)$ und $(x, y + h_2^y)$. Über die Approximation der zweiten Ableitung durch Newtonsche dividierte Differenzen entwickelten Shortley und Weller [SW38] das folgende Differenzenschema

$$\Delta u \simeq \begin{bmatrix} & \dfrac{2}{h_2^y(h_1^y + h_2^y)} & \\ \dfrac{2}{h_1^x(h_1^x + h_2^x)} & -\left(\dfrac{2}{h_1^x h_2^x} + \dfrac{2}{h_1^y h_2^y}\right) & \dfrac{2}{h_2^x(h_1^x + h_2^x)} \\ & \dfrac{2}{h_1^y(h_1^y + h_2^y)} & \end{bmatrix} u. \tag{1.32}$$

Die so erhaltene Diskretisierungsmatrix L_h hat nur noch eine Konsistenzordnung von 1, denn in randnahen Punkten hat $L_h R_h u - R_h \Delta u$ die Größenordnung $O(h)$. Trotzdem liegt Konvergenz der Ordnung 2 vor. Es gilt der folgende

Satz 1.23 *Ω sei beschränkt und in einem Streifen $(x_0, x_0 + d) \times \mathbb{R}$ oder $\mathbb{R} \times (y_0, y_0 + d)$ der Breite d enthalten. Für die zur Shortley-Weller-Diskretisierung von Δu gehörige Matrix L_h gilt dann:*

(i) L_h ist im Allgemeinen nicht symmetrisch;

(ii) L_h ist eine M-Matrix mit $||L_h||^{-1} \leq d^2/8$;

(iii) Das Schema konvergiert von der Ordnung 2.

Für die Approximation der ersten Ableitung ∇u mit dem einfachen Upwind-Verfahren lassen sich die Vorgehensweisen von Shortley und Weller übertragen. Man erhält mit denselben Bezeichnungen wie oben folgende Diskretisierung:

$$b \cdot \nabla u \simeq \begin{bmatrix} & \frac{b_2^-}{h_2^y} & \\ -\frac{b_1^+}{h_1^x} & -\left(\left|\frac{b_1^2}{b_1^+ h_1^x + b_1^- h_2^x}\right| + \left|\frac{b_2^2}{b_2^+ h_1^y + b_2^- h_2^y}\right|\right) & \frac{b_1^-}{h_2^x} \\ & -\frac{b_2^+}{h_1^y} & \end{bmatrix} u. \tag{1.33}$$

Dieses Schema erzielt wie im eindimensionalen Fall eine Konvergenz der Ordnung 1 und wird ebenfalls als einfaches Upwind-Verfahren bezeichnet.

Kapitel 2

Adaptive Verfeinerung

Ein charakteristisches Merkmal für Konvektions-Diffusions Gleichungen sind Bereiche, in denen die Lösung lokal ein fast singuläres Verhalten zeigt. Diese Gebiete bezeichnet man i.A. als *Grenzschicht* oder *layer*. Um die gesuchte Lösung in diesen Bereichen ausreichend genau anzunähern, benötigt man sehr kleine Schrittweiten h. Verfeinert man dazu global das gesamte Gebiet, so bedeutet dies eine Verschwendung von Rechnerkapazität und Speicherressourcen. Weiterhin ist nicht sichergestellt, dass der Diskretisierungsfehler durch sukzessive äquidistante Verfeinerung des Gitters kleiner wird. Deshalb ist es notwendig, in Bereichen um den Layer das Gebiet zu verfeinern. Da die genaue Lage des Layers meistens nicht bekannt ist oder sich bei zeitabhängigen Problemen ändern kann, muss man ein Verfahren entwickeln, welches das Gebiet automatisch bis hin zur gewünschten Genauigkeit verfeinert. Ein solches Verfahren bezeichnet man auch als *adaptives* Verfahren.

Im folgenden Kaptitel soll ein solches Verfahren entwickelt werden. Im ersten Abschnitt wird die Verfeinerungsstrategie für das gegebene Gitter Ω_h vorgestellt. Es wird hierbei auch auf speichertechnische Belange eingegangen. Im zweiten Teil werden die künstlichen Punkte, welche bei der Verfeinerung entstehen, betrachtet.

2.1 Das adaptive Gitter

Ein wesentlicher Aspekt der adaptiven Verfeinerung ist die Entwicklung eines Gitters, welches sich der Beschaffenheit des Problems anpassen kann. Da ein

solches Gitter meistens über keinerlei Struktur mehr verfügt, ist eine spezielle Form der Speicherung notwendig. Die Entwicklung der Rechner und Supercomputer der letzten Jahre hat gezeigt, dass man dabei keinesfalls auf eine Speicherung durch Listen - egal welcher Art - zugreifen sollte. Zum einen ist es schwer, ein Element der Liste direkt anzusprechen, zum anderen sind solche Listen nicht oder zumindest nicht gut parallelisierbar. Gerade der Aspekt der Parallelisierbarkeit hat in den vergangenen Jahren stark an Bedeutung gewonnen. Das liegt vor allem daran, dass die Hardware-Architekturen von einer einzigen "Super-CPU" hin zu verteilten Systemen gewechselt hat. Diese eignen sich hervorragend für parallele Algorithmen und erreichen dabei bemerkenswerte Leistungen.

Betrachten wir zunächst einmal die Verfeinerungsstrategie und leiten dann daraus die notwendige Speicherungstechnik ab. Die vorliegenden Probleme werden allesamt auf dem Einheitsquadrat $\Omega := [0,1]^2$ gelöst. Zu Beginn liegt dort eine äquidistante Punkteverteilung vor, d.h. man hat für gegebenes $h = \frac{1}{n}$, $n \in \mathbb{N}$ ein Teilgebiet $\Omega_0 \subset \Omega$ mit $\Omega_0 := \{(x,y) \in \Omega : \frac{x}{h}, \frac{y}{h} \in \mathbb{Z}\}$. Auf diesem Gitter wird nun die gegebene Differentialgleichung gelöst. Anschließend wird mittels eines Fehlerschätzers geprüft, ob die Lösung der geforderten Güte genügt. An den Stellen, wo dies nicht der Fall ist, wird verfeinert. Dabei geht man so vor, dass ein Quadrat in vier kleinere unterteilt wird. Da es dabei möglich ist, dass nicht jeder neu hinzukommende Punkt sowohl in $X-$ als auch in $Y-$Richtung einen regulären Nachbarpunkt hat, fügt man an den Stellen, wo dies nicht der Fall ist, sogenannte *"künstliche Punkte"* ein (vgl. Abb. (2.1)).

Die künstlichen Punkte liegen in der Mitte eines Quadrates und werden mittels der Eckpunkte dieses Quadrates diskretisiert. Verfeinert man nun weiter, so muss gewährleisten werden, dass dies für jeden künstlichen Punkt gilt. Dazu benötigt man das *Verfeinerungslevel* der einzelnen Quadrate. Zu Beginn hat jedes Quadrat das Verfeinerungslevel 0. Durch jede Unterteilung eines Quadrates in vier kleinere Quadrate erhöht sich das Verfeinerungslevel dieser vier neuen um eins. Um die oben geforderte Struktur des Gitters zu gewährleisten, verlangt man nun, dass je zwei seitlich aneinander grenzende Quadrate sich um höchstens einen Verfeinerungslevel unterscheiden dürfen. Um die geforderte Struktur zu erhalten, kann es passieren, dass durch die Verfeinerung eines Quadrates rekursiv weitere Verfeinerungen erzwungen werden. Abb. (2.2) zeigt einen solchen Fall.

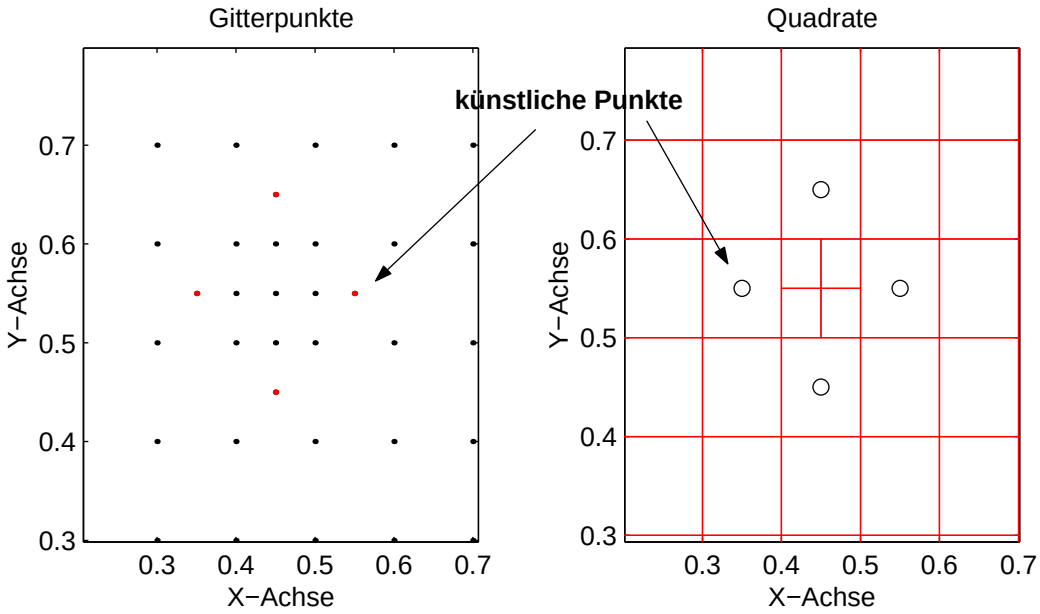


Abbildung 2.1: Verfeinertes Quadrat mit künstlichen Punkten

Verfeinert man derart, so ist gewährleistet, dass jeder Punkt entweder sowohl in $X-$ als auch in $Y-$Richtung einen Nachbarpunkt hat oder aber als künstlicher Punkt im Mittelpunkt eines Quadrates liegt, dessen Eckpunkte aus regulären Punkten bestehen.

Um diese adaptive Verfeinerungsstrategie sinnvoll programmieren zu können, muss man sich überlegen, welche Anforderungen an ein solches Gitter gestellt werden. Man muss zu jedem Punkt seine vier unmittelbaren Nachbarn kennen. Im Falle eines künstlichen Punktes also die vier umliegenden Eckpunkte des Quadrates. Weiterhin muss man zwischen regulären und künstlichen Punkten unterscheiden können. Auch die Quadrate mit ihren Eckpunkten und dem jeweiligen Verfeinerungslevel müssen mit gespeichert werden. Da ein häufiges Zugreifen auf die Punkte notwendig ist, verwendet man zur Speicherung Matrizen.

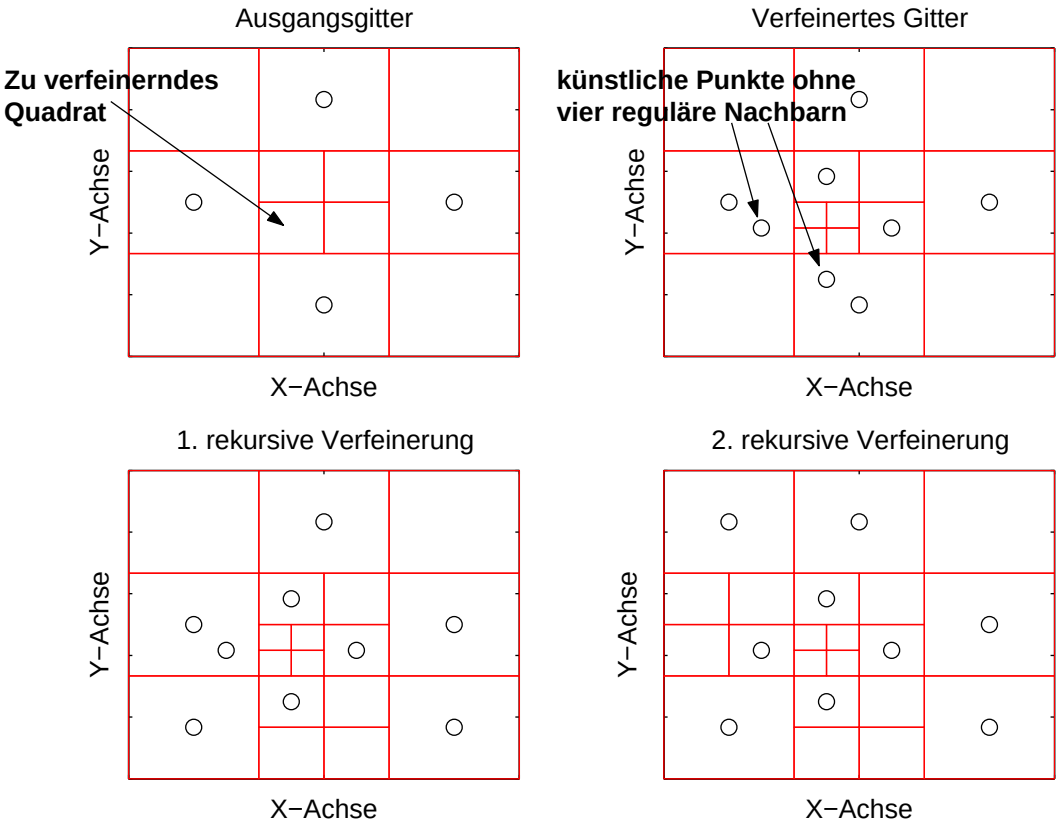


Abbildung 2.2: Verfeinerungsschritt mit künstlichen Punkten

$$P = \begin{pmatrix} x_i \\ y_i \\ \iota_i \in \{0,1\} \end{pmatrix}_{i=1,\ldots,N}$$

In P werden die Punkte des Gitters gespeichert. (x_i, y_i) sind die Koordinaten des Punktes p_i, in ι_i speichert man die Art dieses Punktes: 0 für einen regulären und 1 für einen künstlichen Punkt.

$$E = \begin{pmatrix} p_{i,o} \\ p_{i,r} \\ p_{i,u} \\ p_{i,l} \end{pmatrix}_{i=1,\ldots,N}$$

In E werden die jeweiligen Nachbarn des Punktes in X- bzw. Y-Richtung gespeichert, d.h deren Position in P. Diese werden später zur Diskretisierung der Differentialgleichung benötigt.

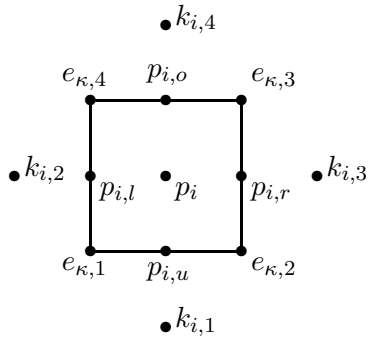
$$Q = \begin{pmatrix} e_{\kappa,1} \\ e_{\kappa,2} \\ e_{\kappa,3} \\ e_{\kappa,4} \\ L_\kappa \end{pmatrix}_{\kappa=1,\ldots,M}$$

In Q werden die Eckpunkte der jeweiligen Quadrate gespeichert und zwar im Uhrzeigersinn beginnend mit der linken unteren Ecke. Gespeichert wird dabei jeweils die entsprechende Position des Punktes in P. $L \in \mathbb{N}_0$ steht für das Level der Verfeinerung des Quadrates.

$$K_P = \begin{pmatrix} x_j \\ y_j \\ \text{Pos. in } P \end{pmatrix}_{j=1,\ldots,K}$$

Die künstlichen Punkte werden zusätzlich gesondert gespeichert.

Mit den so gewählten Bezeichnungen ergibt sich dann für ein zu verfeinerndes Quadrat folgendes Bild:



Die Eckpunkte $e_{\kappa,1}, \ldots, e_{\kappa,4}$ sind auf jeden Fall schon als reguläre Punkte auf dem alten Gitter vorhanden, alle anderen können neu sein. Deswegen muss man für diese Punkte überprüfen, ob sie bereits existieren oder nicht. Dies dient zum einen dazu, keine Punkte doppelt zu speichern, zum anderen kann es durch die Verfeinerung eines Quadrates passieren, dass ein künstlicher Punkt nun durch einen regulären ersetzt wird oder schon als regulärer existiert. Es muss also für ein zu verfeinerndes Quadrat insgesamt neun mal das Vorhandensein dieser Punkte überprüft werden. Dies stellt die Forderung nach einem effizienten Suchalgorithmus auf.

Dazu stellen wir folgende Bedingung an P: alle Punkte in P sind sortiert und zwar aufsteigend zuerst nach ihrer $Y-$Koordinate und anschließend nach ihrer $X-$ Koordinate. Um nun nicht nach jedem neu hinzukommenden Punkt die ganze Matrix P umnummerieren zu müssen, speichert man neue Punkte erst einmal am Ende von P und zusätzlich in einer Matrix N_P. Dies hat den Vorteil, dass man beim Suchen eines Punktes erst einmal in N_P suchen kann. Diese Matrix ist zwar nicht sortiert, da aber dort nur relativ wenige Punkte im Vergleich zu der Anzahl der gesamten Punkte des Gitters gespeichert sind, kann der Suchaufwand als verhältnissmäßig gering erachtet werden. Mit Ausnahme der Punkte $k_{i,1}$ und $k_{i,4}$ liegen alle anderen, falls sie bereits existieren, in der Matrix P zwischen den Punkten $e_{\kappa,1}$ und $e_{\kappa,3}$. D.h., man kann sich beim Suchen auf die dazwischen liegenden Punkte beschränken. Speichert man zusätzlich noch alle künstlichen Punkte gesondert in der Matrix K_P, so kann man wie folgt bei der Suche vorgehen:

(i) suche_Punkt_in(N_P);

(ii) suche_Punkt_in(K_P);

(iii) suche_Punkt_in($P|_{e_{\kappa,1},\ldots,e_{\kappa,3}}$);

Dabei ist suche_Punkt_in(.) eine Funktion, die für vorhandene Punkte ihre Position in P zurückliefert. Die Suche wird abgebrochen, sobald ein Punkt gefunden worden ist. Ist ein Punkt gefunden worden, d.h. war er bereits auf dem alten Gitter vorhanden, so verfährt man wie folgt:

- für reguläre Punkte:
  prüfen, ob sie vorher künstliche Punkte waren;
  falls ja, Punkt in K_P löschen und in P auf regulär ändern.
- für künstliche Punkte:
  keine Änderungen nötig, da sie entweder schon als reguläre Punkte existieren und dann auch weiter als solche behandelt werden, oder aber bereits als künstliche Punkte vorhanden sind, was wiederum keine Änderung erfordert.

Ist ein Punkt nicht gefunden worden, d.h. er kommt neu hinzu, so geht man auf folgende Weise vor:

- Punkt hinten an P anhängen und merken, ob er regulär oder künstlich ist.
- Punkt hinten an N_P anhängen und zusammen mit der Position in P speichern.
- Handelt es sich um einen künstlichen Punkt, so wird er zusätzlich am Ende von K_P gespeichert.

Hat man nun alle Punkte entsprechend in den Matrizen gespeichert, so müssen die Kantenverweise in der Matrix E angeglichen werden. Die Eckpunte $e_{\kappa,1},\ldots,e_{\kappa,4}$ und die Punkte $p_{i,l}, p_{i,r}, p_{i,u}, p_{i,o},$ und p_i sind auf jeden Fall reguläre Punkte und ihre direkten Nachbarn sind bekannt. Die Kanten dieser Punkte können also sofort in der Matrix E aktualisiert werden. Anders verhält es sich mit den künstlichen Punkten $k_{i,1},\ldots,k_{i,4}$. Es können hierbei zwei Fälle auftreten:

(i) der Punkt existierte bereits als regulärer oder künstlicher Punkt, oder

(ii) der Punkt ist ein neuer künstlicher Punkt.

Im ersten Fall ist keine Änderung in E notwendig, da nach Voraussetzung alle Punkte des alten Gitters vollständig gespeichert sind. Im zweiten Fall bleibt zu prüfen, ob der neue künstliche Punkt in der Mitte eines Quadrates liegt. In diesem Fall werden die Eckpunkte dieses Quadrates in E gespeichert und zwar $E(k_{i,j}) = [e_{\tau,3}, e_{\tau,2}, e_{\tau,1}, e_{\tau,4}]^T,\ j \in \{1, \ldots, 4\}$. Mit diesen Eckwerten wird später der Wert für den künstlichen Punkt interpoliert.

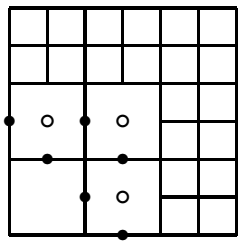
Liegt der neue künstliche Punkt nicht in der Mitte eines Quadrates, so erzwingt er die Verfeinerung des Quadrates, in dem er liegt (vgl. nochmals Abb. (2.2)). Die Kanten bzw. Ecken dieses Punktes können also nicht sofort gespeichert werden, sondern müssen so lange offen bleiben, bis die Rekursion wieder auf der Ebene angelangt ist, von der sie aufgerufen wurde. Erst dann können die zur späteren Interpolation notwendigen Eckpunkte gespeichert werden.

Ein weiterer wichtiger Punkt ist, dass die rekursive Verfeinerung kein Quadrat verfeinert, welches auch durch den Fehlerschätzer von vorne herein verfeinert worden wäre. Dabei könnte es nämlich passieren, dass ein Quadrat öfter verfeinert wird, als notwendig. Dies verhindert man durch Sortierung der durch den Fehlerschätzer zur Verfeinerung vorgesehenen Quadrate nach dem Level ihrer Verfeinerung. Wie man sich leicht überlegt, kann ein Quadrat durch seine Unterteilung in vier kleinere Quadrate nur die rekursive Verfeinerung solcher Quadrate erzwingen, die ein gröberes Feinheitslevel haben als es selber. Sortiert man nun die zu verfeinernden Quadrate aufsteigend nach ihrem Verfeinerungslevel L und beginnt mit dem gröbsten, so bleiben alle anderen (feineren oder gleichgroben) Quadrate davon unberührt.

Die Methode der Unterteilung der zu verfeinernden Quadrate bietet auch eine Ansatzmöglichkeit zur Parallelisierung der Verfeinerung. Allerdings muss man dazu Quadrate zu Klassen zusammenfassen, die auf jeden Fall bzgl. ihrer Verfeinerung disjunkt sind. D.h., dass es kein Quadrat gibt, welches durch zwei verschiedene Klassen von Quadraten zur Verfeinerung gezwungen wird. Dies ist höchstwahrscheinlich mit der bisher verwendeten Methode der Verfeinerung nicht zu erreichen. Man muss evtl. die Strategie weiter verfeinern, indem man z.B. die Quadrate zuerst nur in eine Richtung verfeinert.

2.2 Künstliche Punkte

In diesem Abschnitt wollen wir uns mit der Handhabung der künstlichen Punkte, welche durch die adaptive Verfeinerung des Gitters Ω_h entstehen, beschäftigen. Diskretisiert man diese Punkte mit einem 5-Punkt-Stern, so muss man evtl. fehlende Nachbarpunkte auf den Kanten durch Interpolation über Eckpunkte bestimmen. Das folgende Bild zeigt den Ausschnitt eines verfeinerten Gitters mit künstlichen Punkten $\circ$. Die Hilfspunkte $\bullet$ auf den Kanten sind nicht bekannt und müssen über Interpolation bestimmt werden.



Dabei ist quadratische Interpolation die minimalste Anforderung, wenn man global eine Genauigkeit der Ordnung zwei erreichen will. Da der Laplace-Operator die zweite Ableitung einer Funktion beinhaltet, treten bei seiner Diskretisierung Divisionen durch h^2 auf. Das bedeutet, dass für eine Interpolation der Ordnung $O(h^p)$ durch die Division mit h^2 aus der zweiten Ableitung man insgesamt nur eine globale Ordnung von $O(h^{p-2})$ erhält. Für eine quadratische Interpolation ($p = 3$) erhält man also immer noch einen Fehler (*truncation error*) von $O(h)$ an dem Übergang vom feinen zum groben Gitter (vgl. [AN98]).

Um dieses Problem zu umgehen, führt man an den künstlichen Punkten einen um 45° gedrehten Stern ein und diskretisiert sie mit den Eckpunkten der Quadrate, in denen sie liegen. Man definiert die Koordinatentransformation

$$\left.\begin{aligned} \xi &= \tfrac{1}{2}\sqrt{2}(x+y), \\ \eta &= \tfrac{1}{2}\sqrt{2}(x-y). \end{aligned}\right\} \tag{2.1}$$

Damit erhält man

$$\begin{aligned} u_{xx} + u_{yy} &= u_{\xi\xi} + u_{\eta\eta}, \\ u_x &= \frac{1}{2}\sqrt{2}(u_\xi + u_\eta), \\ u_y &= \frac{1}{2}\sqrt{2}(u_\xi - u_\eta). \end{aligned}$$

Hiermit ergibt sich dann für die ursprüngliche Differentialgleichung

$$-\varepsilon(u_{xx} + u_{yy}) + b_1 u_x + b_2 u_y + cu = f$$

die transformierte Gleichung

$$-\varepsilon(u_{\xi\xi} + u_{\eta\eta}) + \frac{1}{2}\sqrt{2}(b_1 + b_2)u_\xi + \frac{1}{2}\sqrt{2}(b_1 - b_2)u_\eta + cu = f, \tag{2.2}$$

mittels derer man die künstlichen Punkte diskretisiert. Damit erhält man für den transformierten Laplace-Operator das Schema

$$-\varepsilon\Delta u(\xi,\eta) \simeq \frac{1}{\tilde{h}^2}\begin{bmatrix} -1 & & -1 \\ & 4 & \\ -1 & & -1 \end{bmatrix} u(\xi,\eta). \tag{2.3}$$

Hierbei bezeichnet $\tilde{h}$ die Hälfte der Diagonale des Quadrates, in dem der künstliche Punkt liegt. Für die so transformierte Differentialgleichung lässt sich nun auch das stabilisierende einfache Upwind Verfahren formulieren. Man definiert

$$\beta_1 := \frac{1}{2}\sqrt{2}(b_1 + b_2), \tag{2.4}$$

$$\beta_2 := \frac{1}{2}\sqrt{2}(b_1 - b_2) \tag{2.5}$$

und erhält damit

$$\beta_1 u_\xi + \beta_2 u_\eta \simeq \frac{1}{\tilde{h}}\begin{bmatrix} -\beta_2^+ & & \beta_1^- \\ & |\beta_1| + |\beta_2| & \\ -\beta_1^+ & & \beta_2^- \end{bmatrix} u(\xi,\eta), \tag{2.6}$$

mit $\beta_i^+ := \max\{0, \beta_i\}$, $\beta_i^- := \min\{\beta_i, 0\}$. Mit den so erhaltenen Differenzenschemata lassen sich nun die künstlichen Punkte ohne Interpolation fehlender Punkte diskretisieren.

Kapitel 3

Fehlerschätzer

Wie bereits in der Einleitung erwähnt, besitzen singulär gestörte Differentialgleichungen Stellen, an denen die Lösung sich nahezu unstetig verhält. Diese Stellen bezeichnet man auch als *Layer*. Versucht man nun, diese Differentialgleichungen an diskreten Punkten zu lösen, so ist es notwendig, an diesen Stellen das Gitter feiner zu wählen als an den übrigen. Da i.A. die Lage des Layers nicht von vorneherein bekannt ist bzw. sich bei zeitabhängigen Problemen im Laufe der Zeit ändern kann, ist es notwendig, ein Verfahren zu entwickeln, welches die Verfeinerung automatisch bis hin zu einer gewünschten Genauigkeit vornimmt.

In dem nun folgenden Kapitel wird ein auf residuen-basierender Fehlerschätzer hergeleitet. Dieser soll den Fehler jeweils zwischen zwei Punkten (1D) bzw. in der Mitte eines Quadrates (2D) schätzen. Der dritte Teil dieses Kapitels beschreibt einen von [AN98] vorgestellten Fehlerschätzer. Die dort erzielten Ergebnisse werden später mit den unsrigen verglichen.

3.1 Der eindimensionale Fall

Es werden zunächst einige wichtige Ergebnisse des eindimensionalen Falls zusammengefasst. Dazu betrachtet man die Randwertaufgabe

$$\left.\begin{array}{rcl} -\varepsilon u'' + u' & = & 0 \quad \text{in} \quad \Omega = (0,1), \\ u(0) = 0 \quad , & & u(1) = 1. \end{array}\right\} \tag{3.1}$$

Dies stellt einen speziellen Fall der Konvektions-Diffusions-Gleichung ($b \equiv 1, c \equiv 0$) dar. Da die Singularität im Wesentlichen nicht von dem Term $c(x) \cdot u$ abhängt, kann man diesen vernachlässigen. Der Fall $b \equiv 1$ stellt ebenfalls keine unzulässige Einschränkung dar. Die Koeffizientenfunktion $b(x)$ des Konvektionsterm ist stetig und beschränkt auf Ω. Der allgemeinere Fall lässt sich also durch die Wahl entsprechender Konstanten C abschätzen. Durch die Transformation $x \mapsto -x$ lassen sich alle Überlegungen auch für den Fall $b \leq 0$ übertragen. Für den Fall, dass b das Vorzeichen wechselt, muss man das Gebiet in Teilgebiete mit $b(x) > 0$ bzw. $b(x) < 0$ aufteilen.

Es sei

$$\begin{aligned} u &: \text{exakte Lösung von (3.1)} \\ \tilde{u}_h &: \text{Näherung mittels upstream Verfahrens 1. Ordnung} \\ I_h &: \text{Restriktion auf das Gitter } \Omega_h \end{aligned}$$

Es bezeichne $\Omega_h := I_h\Omega = \{x_0, x_1, \ldots, x_n\}$ die Menge der diskreten Punkte auf Ω. Man definiert das Residuum

$$r_h := L_h\tilde{u}_h - f_h \,, \tag{3.2}$$

mit der Iterationsmatrix L_h (mit zentralen Differenzen) und $f_h = I_h(f)$, die auf Ω_h eingeschränkte "rechte Seite" von (3.1). Für alle Punkte des Gitters Ω_h und für alle Zwischenpunkte $\frac{1}{2}(x_j + x_{j+1})$ $(j = 0, \ldots, n-1)$, an denen der Fehler berechnet werden soll, definiert man dann damit

$$\frac{r_{h,i}}{|L_{h,i}|_1} =: \sigma_i. \tag{3.3}$$

Nun bleibt zu zeigen, daß σ_i ein geeigneter Schätzer ist. Es gilt

$$\begin{aligned} r_h &= L_h\tilde{u}_h - f_h \\ &= L_h\tilde{u}_h - L_hI_hu + L_hI_hu - I_hLu \,. \end{aligned}$$

Damit erhält man

$$\begin{aligned} L_h(\tilde{u}_h - I_hu) &= r_h + \tau_h \,, \text{ mit} \\ \tau_h &:= (I_hL - L_hI_h)u \quad \text{"truncation error"}. \end{aligned} \tag{3.4}$$

Zur Abschätzung von τ_h benötigt man das folgende Lemma:

Lemma 3.1 *Für* $[x-h, x, x+h] \subset \bar{\Omega}$ *sei*

$$\begin{aligned} \delta^0 u(x) &:= \frac{1}{2h}(u(x+h) - u(x-h)) \quad , \\ \delta^- \delta^+ u(x) &:= \frac{1}{h^2}(u(x-h) - 2u(x) + u(x+h)) \quad . \end{aligned}$$

Dann gilt jeweils:

$$\delta^0 u(x) = u'(x) + h^2 R \text{ mit } |R| \leq \frac{1}{6}||u||_{C^3}, \text{ falls } u \in C^3(\bar{\Omega}) \tag{3.5}$$

$$\delta^- \delta^+ u(x) = u''(x) + h^2 R \text{ mit } |R| \leq \frac{1}{12}||u||_{C^4}, \text{ falls } u \in C^4(\bar{\Omega}) \tag{3.6}$$

Beweis: [Ha86].

Daraus folgt

$$\begin{aligned} |\tau_h| &\leq \varepsilon \frac{h^2}{12}|u^{\mathrm{iv}}| + \frac{h^2}{6}|u'''| \\ &\leq (\frac{\varepsilon}{12}h^2 + \frac{1}{6}h^2) \max\{|u^{\mathrm{iv}}|_\infty, |u'''|_\infty\}. \end{aligned} \tag{3.7}$$

Mit $e_h := \tilde{u}_h - I_h u$ und (3.4) erhält man für den i-ten Knotenpunkt des Gitters

$$\frac{(L_h e_h)_i}{|L_{h,i}|_1} = \underbrace{\frac{r_{h,i}}{|L_{h,i}|_1}}_{=\sigma_i} + \frac{\tau_{h,i}}{|L_{h,i}|_1}. \tag{3.8}$$

Es bleibt nun zu zeigen, daß $e_i \leq \sigma_i$ gilt. Dazu unterscheidet man zwei Fälle

(i) $\varepsilon << h$ (weg vom Layer),

(ii) $h << \varepsilon$ (im Layer-Bereich).

<u>(i) $\varepsilon << h$:</u>

$$\begin{aligned} |L_{h,i}|_1 &= \frac{4\varepsilon}{h_i^2} + \frac{1}{h_i} \approx \frac{1}{h_i} \\ \Longrightarrow \frac{\tau_{h,i}}{|L_{h,i}|_1} &= \frac{O(h_i^2)}{O\left(\frac{1}{h_i}\right)} = O(h_i^3) << 1. \\ \Longrightarrow \frac{(L_h e_h)_i}{|L_{h,i}|_1} &\approx \sigma_i. \end{aligned}$$

Andererseits erhält man durch Einsetzen des Diskretisierungsoperators:

$$\begin{aligned}
\frac{(L_h e_h)_i}{|L_{h,i}|_1} &= \frac{(-\varepsilon\delta^-\delta^+ e_h + \delta^0 e_h)_i}{\frac{4\varepsilon}{h_i^2} + \frac{1}{h_i}} \\
&\overset{\text{Lemma(3.1)}}{=} \frac{-\varepsilon(e_i'' + C_1 h_i^2 e_i^{(4)}) + e_i' + C_2 h_i^2 e_i'''}{\frac{4\varepsilon}{h_i^2} + \frac{1}{h_i}} \\
&\simeq \frac{-\varepsilon e_i'' + e_i'}{\frac{4\varepsilon}{h_i^2} + \frac{1}{h_i}} \\
&\overset{\varepsilon\to 0}{\simeq} \underbrace{h_i e_i'}_{\text{lokaler Fehleranstieg in } x_i} \approx \sigma_i.
\end{aligned}$$

Damit erhält man schließlich

$$\begin{aligned}
e_i &\approx e_{i-1} + h_{i-1} e_{i-1}' \\
&\approx \underbrace{e_{i-1}}_{\text{alter Fehler}} + \underbrace{\sigma_{i-1}}_{\text{lok. Fehlerzuwachs}}.
\end{aligned}$$

(ii) $h << \varepsilon$:

$$\begin{aligned}
\frac{\tau_{h,i}}{|L_{h,i}|_1} &\leq \frac{\varepsilon\frac{h_i^2}{12}|u^{(4)}| + \frac{h_i^2}{6}|u'''|}{\frac{4\varepsilon}{h_i^2} + \frac{1}{h_i}} \\
&= \frac{\varepsilon\frac{h_i^4}{12}|u^{(4)}| + \frac{h_i^4}{6}|u'''|}{4\varepsilon + h_i} \\
&\overset{h_i<<\varepsilon}{\approx} \frac{h_i^4}{48}|u_i^{(4)}| + \frac{h_i^4}{24\varepsilon}|u_i'''|.
\end{aligned}$$

Da die exakte Lösung von (3.1) die Form

$$u(x) = \frac{\exp(\frac{x-1}{\varepsilon}) - \exp(-\frac{1}{\varepsilon})}{1 - \exp(-\frac{1}{\varepsilon})}$$

hat und somit $|u^{\mathrm{iv}}| \simeq \frac{1}{\varepsilon^4}$ und $|u_i'''| \simeq \frac{1}{\varepsilon^3}$ gilt, ergibt sich zusammen mit $h << \varepsilon$

$$\frac{\tau_{h,i}}{|L_{h,i}|_1} << 1.$$

Man hat also auch für diesen Fall

$$\frac{(L_h e_h)_i}{|L_{h,i}|_1} \approx \sigma_i.$$

Andererseits gilt

$$\begin{aligned}\frac{(L_h e_h)_i}{|L_{h,i}|_1} &= \frac{-\frac{\varepsilon}{h_i^2}-\frac{1}{2h_i}}{\frac{4\varepsilon}{h_i^2}+\frac{1}{h_i}}\,e_{i-1}+\frac{\frac{2\varepsilon}{h_i^2}}{\frac{4\varepsilon}{h_i^2}+\frac{1}{h_i}}\,e_i+\frac{-\frac{\varepsilon}{h_i^2}+\frac{1}{2h_i}}{\frac{4\varepsilon}{h_i^2}+\frac{1}{h_i}}\,e_{i+1}\\ &\approx -\frac{1}{4}e_{i-1}+\frac{1}{2}e_i-\frac{1}{4}e_{i+1}.\end{aligned}$$

Daraus ergibt sich also

$$\begin{aligned}\sigma_i &\approx -\frac{1}{4}e_{i-1}+\frac{1}{2}e_i-\frac{1}{4}e_{i+1}\\ \Longrightarrow e_i &\approx \underbrace{2\sigma_i}_{\text{lok. Fehlerzuwachs}}+\underbrace{\frac{1}{2}e_{i-1}+\frac{1}{2}e_{i+1}}_{\text{alte Fehler}}.\end{aligned}$$

Zusammenfassend kann man also den Fehlerschätzer σ_i als Maß des lokalen Anwachsens des Fehlers auffassen. Dabei wird der Schätzer jeweils an Punkten zwischen zwei Gitterpunkten aus dem vorherigen Verfeinerungsschritt ausgewertet, d.h. $x_i = \frac{1}{2}(x_{i-1} + x_{i+1})$. Der Wert für die Lösung u an der Stelle $u_i = u(x_i)$ wird mittels Interpolation durch die beiden Nachbarpunkte ermittelt, also $u_i \approx \frac{1}{2}(u_{i-1} + u_{i+1})$.

Diskretisiert man nun an diesem Punkt die zweite Ableitung u'' durch zentrale Differenzen, so erhält man

$$\begin{aligned}-u_i'' &\simeq \frac{1}{h^2}\left[\,-1,\quad 2,\quad -1\,\right]u_i\\ &= \frac{1}{h^2}\left\{-u_{i-1}+2\left(\frac{1}{2}\left(u_{i-1}+u_{i+1}\right)\right)-u_{i+1}\right\}\\ &= 0.\end{aligned}$$

Für den ersten Fall ($\varepsilon << h$) stellt dieses Resultat kein Problem dar, da der Anteil der zweiten Ableitung für $\varepsilon \to 0$ sowieso nicht in die Schätzung miteinfließt. Für den zweiten Fall hingegen muss man eine gesonderte Betrachtung durchführen. Es gilt

$$\begin{aligned}\frac{\frac{\varepsilon}{h^2}\left[\,-1,\quad 2,\quad -1\,\right]}{|L_{h,i}|_1} &\approx \frac{-C\varepsilon u_i''+O(h^2)}{\frac{\varepsilon}{h^2}+\frac{1}{h}}\\ &\leq -Ch^2u_i''+O(h^2).\end{aligned}$$

Falls $u \in C^2$, ist $u'' = O(\frac{1}{\varepsilon^2})$, für $\varepsilon \to 0$. Damit erhalten wir schließlich

$$\begin{aligned} \frac{\frac{\varepsilon}{h^2}\left[\begin{array}{ccc} -1, & 2, & -1 \end{array}\right]}{|L_{h,i}|_1} &\leq C'\frac{h^2}{\varepsilon^2} \\ &= O(h^2) << 1. \end{aligned} \tag{3.9}$$

Dies rechtfertigt auch für den zweiten Fall die oben durchgeführten Rechnungen.

Wir haben also gesehen, dass für die beiden wesentlichen Fälle ($h << \varepsilon$ bzw. $\varepsilon << h$) sich der Schätzer zumindest wie der Fehler verhält. Zwar liefert uns dieser residuenbasierende Schätzer keinen wirklichen Fehlerschätzer, aber σ_i wird klein, falls der Fehler klein wird. Dies genügt, wie wir später auch noch anhand der numerischen Ergebnisse sehen werden, den Anforderungen, ein "optimales" Gitter zu finden.

Führt man obige Rechnungen mit $|L_{h,i}|_\infty$ durch, so erhält man auf analoge Art und Weise für die beiden Fälle

$$\begin{aligned} h << \varepsilon &: |e_i| \leq \sigma_i + \frac{1}{2}|e_{i-1}| + \frac{1}{2}|e_{i+1}| \\ \varepsilon << h &: \sigma_i \to \frac{1}{2}\sigma_i. \end{aligned}$$

3.2 Der zweidimensionale Fall

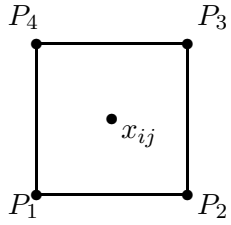
Die Ergebnisse des eindimensionalen Falls werden nun auf den zweidimensionalen Fall übertragen. Hierbei kann der 2D-Fall als Erweiterung des 1D-Falls interpretiert werden, indem Ableitungen in x-Richtung mit Ableitungen in y-Richtung mittels Tensorprodukten kombiniert werden. Dies rechtfertigt die Annahme, dass sich die Ergebnisse für den Fehlerschätzer aus dem Eindimensionalen analog auf den zweidimensionalen Fall übertragen lassen.

Es wird die Randwertaufgabe

$$\left.\begin{array}{rcll} Lu & := & -\varepsilon\Delta u + b_1 u_x + b_2 u_y + cu = f & u \in \Omega = (0,1)^2 \\ u & = & B & u \in \partial\Omega \end{array}\right\} \tag{3.10}$$

betrachtet. Mit L_h bezeichnet man den zentralen Differenzen-Operator für (3.10) und mit $L_h^{(i,j)}$ den diskreten Operator an der Stelle $x_{ij} := (x_i, y_j) \in \Omega$.

Der Fehler soll jeweils in der Mitte eines Quadrates geschätzt werden. Die Näherung $\tilde{u}$ der Lösung u ist an den Eckpunkten des Quadrates aus dem vorherigen Iterationsschritt bekannt.



Die Gleichung (3.10) wird mittels

$$\begin{aligned} \xi &= \tfrac{1}{2}\sqrt{2}(x+y) \\ \eta &= \tfrac{1}{2}\sqrt{2}(x-y) \end{aligned}$$

auf

$$-\epsilon(u_{\xi\xi}+u_{\eta\eta})+\frac{1}{2}\sqrt{2}(b_1+b_2)u_\xi+\frac{1}{2}\sqrt{2}(b_1-b_2)u_\eta+cu=f \tag{3.11}$$

transformiert. Die Ableitungen werden durch zentrale Differenzen ausgewertet, d.h.

$$-\varepsilon\Delta u(\xi,\eta) \approx \frac{\varepsilon}{h^2}\begin{bmatrix} -1 & & -1 \\ & 4 & \\ -1 & & -1 \end{bmatrix},$$

$$\underbrace{\frac{1}{2}\sqrt{2}(b_1+b_2)}_{:=\beta_1} u_\xi \approx \frac{\beta_1}{h}\begin{bmatrix} 0 & & 1 \\ & 0 & \\ -1 & & 0 \end{bmatrix},$$

$$\underbrace{\frac{1}{2}\sqrt{2}(b_1-b_2)}_{:=\beta_2} u_\eta \approx \frac{\beta_2}{h}\begin{bmatrix} -1 & & 0 \\ & 0 & \\ 0 & & 1 \end{bmatrix}.$$

Daraus ergibt sich dann

$$L_h^{(\xi,\eta)} = \begin{bmatrix} -\frac{\varepsilon}{h^2}-\frac{\beta_2}{h} & & -\frac{\varepsilon}{h^2}+\frac{\beta_1}{h} \\ & \frac{4\varepsilon}{h^2}+c & \\ -\frac{\varepsilon}{h^2}-\frac{\beta_1}{h} & & -\frac{\varepsilon}{h^2}+\frac{\beta_2}{h} \end{bmatrix}.$$

Der Wert $\tilde{u}_{ij}$ in der Mitte des Quadrates wird durch die Eckpunkte $P_1, \ldots, P_4$ interpoliert:

$$\tilde{u}_{ij} \approx \frac{1}{4} \sum_{i=1}^{4} \tilde{u}(P_i) \quad .$$

Damit lässt sich nun das Residuum der transformierten Gleichung in der Mitte des Quadrates bestimmen:

$$r_h^{ij} = L_h^{(i,j)} \tilde{u} - f_{ij}. \tag{3.12}$$

Analog zum eindimensionalen Fall definiert man nun als Fehlerschätzer

$$\sigma_{ij} := r_h^{(i,j)} \cdot ||L_h^{(i,j)}||_1^{-1}, \tag{3.13}$$

wobei die Gewichtung mit der Norm des zentralen Differenzenoperators $||L_h^{(i,j)}||_1$ eine Äquilibrierung erzielt.

3.3 Fehlerschätzer von Axelsson

Im folgenden Abschnitt wird ein Verfahren zur adaptiven Verfeinerung von [AN98] vorgestellt. Die dort erzielten Ergebnisse sollen mit den eigenen Resultaten verglichen werden. Leider scheiterte der Versuch, dass von Axelsson und Nikolova vorgestellte Verfahren erfolgreich zu implementieren, so dass ein Vergleich nur anhand der in [AN98] veröffentlichten Werte erfolgen konnte.

Axelsson und Nikolova führen in ihrer Arbeit eine Defekt-Korrektur ein, um eine höhere Genauigkeit der diskreten Lösung zu erlangen. Dabei verwenden sie einen instabilen Operator $L_h^{(0)}$ von hoher Ordnung und einen stabilen Operator $L_h^{(1)}$ von niedriger Ordnung. Für $L_h^{(1)}$ wählen sie einen Operator erster Ordnung, welcher Δu mit zentralen Differenzen und u_x, u_y mit Upstream erster Ordnung approximiert. $L_h^{(0)}$ ist ein Operator zweiter Ordnung, der zentrale Differenzen sowohl für Δu als auch für u_x, u_y verwendet.

Die Defekt-Korrektur wird dann wie folgt definiert:

(i) Löse $L_h^{(1)} u_h^{(1)} = f_h$,

(ii) Löse $L_h^{(1)} \delta_h = f_h - L_h^{(0)} u_h^{(1)}$ und setze $u_h = u_h^{(1)} + \delta_h$.

Damit ergibt sich dann für den Diskretisierungsfehler

$$||e_h|| = ||u - u_h^{(1)} + u_h^{(1)} - u_h|| = ||u - u_h^{(1)} - \delta_h|| = ||(L_h^{(1)})^{-1}\tau_h^{(1)} - \delta_h||,$$

wobei

$$\tau_{h,ij}^{(1)} = \begin{cases} -\frac{h_{ij}}{2}\left(|b_1|\frac{\partial^2 u}{\partial x^2} + |b_2|\frac{\partial^2 u}{\partial y^2}\right)_{ij} + o(h_{ij}) & \text{falls } (x_i, y_j) \text{ regulärer Punkt,} \\ -\frac{h_{ij}}{2}\left(|b_1|\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + 2|b_2|\frac{\partial^2 u}{\partial x \partial y}\right)_{ij} + o(h_{ij}) & \text{falls } (x_i, y_j) \text{ künstlicher Punkt} \end{cases}$$

der Abschneidefehler des Upwind-Operators ist. Demzufolge sind die führenden Terme $\hat{\tau_h}^{(1)}$ von $\tau_h^{(1)}$ die zweiten Ableitungen und man kann sie approximieren, indem man für jeden Punkt zentrale Differenzen verwendet. Man löst also $L_h^{(1)}\hat{e_h}^{(1)} = \hat{\tau_h}^{(1)}$ und erhält als Fehlerschätzer

$$e_h^{appr} = \hat{e_h}^{(1)} - \delta_h.$$

Alle Punkte, an denen e_h^{appr} größer oder gleich einer gegebenen Toleranz ist, werden verfeinert. Die dabei entstehenden Verfeinerungen des ursprünglichen Gebietes Ω dürfen nicht überlappend sein und ihr Rand besteht aus achsenparallelen minimalen Polygonen, die die zu verfeinernden Punkte einschließen. Bezeichnet man mit $k \in \{0, \ldots, k_{max}\}$ das Verfeinerungslevel, wobei Level 0 das gröbste Level ist und das ganze Gebiet umfasst, so besteht das Gitter Ω_k aus der Vereinigung der verfeinerten Teilgebiete für das Level k. Das bedeutet:

$$\begin{aligned} &\Omega_k \subset \Omega_{k-1} \subset \Omega_{k-2} \subset \ldots \subset \Omega_0, \\ &\Omega_k = \bigcup \Omega_k^l, \quad l = 1, 2, \ldots, \quad \Omega_k^l \cap \Omega_k^{l+1} \in \{\emptyset\}, \\ &\partial\Omega_k = \bigcup \partial\Omega_k^l, \quad l = 1, 2, \ldots, \quad \partial\Omega_k^l \cap \partial\Omega_k^{l+1} \in \{\emptyset, \text{isolierte Punkte}\}. \end{aligned}$$

Zu beachten ist hierbei, dass die einzelnen Verfeinerungslevel ineinander geschachtelt sein müssen, d.h. es wird keine nachträgliche Verfeinerung eines gröberen Levels zu einem späteren Zeitpunkt zugelassen.

Im Rahmen dieser Arbeit wurde der Versuch unternommen, die Vorgehensweise in [AN98] auf die eigene Verfeinerungsstrategie zu übertragen, um so einen Vergleich der Fehlerschätzer anstellen zu können. Dabei wurde die in Kapitel (2) eingeführte Verfeinerungsstrategie verwendet. Die Defekt-Korrektur und der Fehlerschätzer sind wie in [AN98] übernommen worden.

Leider erwies sich die Defekt-Korrektur als instabil, so dass das Verfahren nicht konvergierte. Bereits nach dem ersten Iterationsschritt in der Defekt-Korrektur wuchs der Fehler $|u - u_h|$ drastisch an. Je mehr Iterationsschritte

ausgeführt wurden, desto größer wurde er. Demzufolge lieferte auch der Fehlerschätzer von Axselsson und Nikolova keine brauchbaren Ergebnisse, da die Lösung aus der Defekt-Korrektur dort einfließt.

Um zu testen, ob die Defekt-Korrektur auf dem feinsten Gitter eine Verbesserung der Lösung bewirkt, wurde diese auf einem verfeinerten Gitter getestet, welches mittels des eigenen Fehlerschätzers nach der Konvergenz des adaptiven Algorithmus berechnet worden war. Auch hier wuchs der Fehler bereits nach dem ersten Iterationsschritt drastisch an und wurde größer, je mehr Iterationsschritte ausgeführt wurden.

Diese Ergebnisse lassen nur den Schluss zu, dass die Defekt-Korrektur zumindest bei dieser Art der Verfeinerung nicht funktioniert. Der instabile Operator $L_h^{(0)}$ kann zur Verwendung der Defektberechnung nicht ohne besondere Behandlung verwendet werden.

Kapitel 4

Löser und Postprocessing

Bei der Diskretisierung von partiellen Differentialgleichungen entstehen lineare Gleichungssysteme, die möglichst effizient gelöst werden sollen. D.h. möglichst hohe Genauigkeit bei möglichst geringem Aufwand. Das lineare Gleichungssystem

$$Ax = b, \tag{4.1}$$

welches bei finiten Differenzen auftritt, enthält eine dünnbesetzte Matrix A. Zur Lösung werden zum einen das GMRES-Verfahren und zum anderen das Packet UMFPACK [Da04] für die *sparse*-LU-Zerlegung verwendet. Zur Lösung von Gleichungssystemen, deren Anzahl der Unbekannten eine direkte Lösung zulässt, ist letzteres Verfahren sehr gut geeignet. Bei größer werdenden Systemen ist ein direkter Löser aufgrund der Speicherbeschränkung häufig nicht anwendbar. In diesem Fall wird hier das GMRES-Verfahren verwendet. Dabei wird ausgenutzt, dass dünnbesetzte Matrizen bei einer Matrix-Vektor-Multiplikation nur $O(n)$ Operationen benötigen.

In den ersten beiden Abschnitten dieses Kapitels werden beide Verfahren zur Lösung der linearen Gleichungssysteme vorgestellt. Der dritte Abschnitt befasst sich mit der Entwicklung eines Postprocessings, welches auf dem adaptiv verfeinerten Gitter eine größere Genauigkeit der gesuchten Lösung liefern soll.

4.1 Lösen der Gleichungssysteme

4.1.1 *sparse*-LU-Zerlegung mit UMFPACK

UMFPACK behandelt die direkte Lösung von linearen Gleichungssystemen $Ax = b$, wobei A dünnbesetzt und unsymmetrisch ist. Es handelt sich hierbei um eine Methode, welche Spaltenvorsortierungen mit rechtsseitigen unsymmetrischen multifrontalen Faktorisierungen kombiniert.

UMFPACK faktorisiert Matrizen $PAQ, PRAQ$ oder $PR^{-1}AQ$ in das Produkt LU, wobei L und U untere bzw. obere Dreiecksmatrizen sind. P und Q sind Permutationsmatrizen und R ist eine Diagonalmatrix mit Zeilenskalierungsfaktoren. Beide Matrizen P und Q werden so gewählt, dass Fill-in (neue Nicht-Null-Einträge in L und U, die nicht in A vorhanden waren) reduziert wird. Dabei hat P die zusätzliche Rolle, mittels relaxierter partieller Pivotwahl numerische Stabilität zu sichern. Die Spaltenvorsortierung Q wird hierbei so gewählt, dass eine gute obere Schranke für das Fill-in gegeben wird. Diese Wahl ist unabhängig von der Zeilensortierung P.

Die beste Sortierung zu finden, ist ein NP-vollständiges Problem [Ya81]. Aus diesem Grund verwendet UMFPACK heuristische Ansätze. Die Matrix wird analysiert und es wird eine der drei Strategien zum Vorsortieren der Zeilen und Spalten ausgewählt: *unsymmetrisch,* 2×2 und *symmetrisch.* Diese Strategien werden unten beschrieben.

Zuerst werden alle Pivots mit Null-Markovitzkosten eliminiert und in die LU-Faktoren geschrieben. Dies sind Pivots, dessen Pivotzeile oder Pivotreihe (oder beide) nur einen Nicht-Null-Eintrag haben und demnach ohne Fill-in zu erzeugen ersetzt werden können. Die verbleibende Untermatrix S wird anschließend analysiert. Falls die Nicht-Null-Einträge der Matrix S sehr unsymmetrisch sind, wird die unsymmetrische Strategie verwendet. Falls die Einträge nahezu symmetrisch sind und die Matrix eine nullfreie Diagonale hat, wird die symmetrische Strategie verwendet. Ansonsten wird die 2×2-Strategie angewandt. Dabei wird eine Zeilenpermutation P_2 gefunden, welche versucht, die Anzahl der kleinen Diagonaleinträge von P_2S zu reduzieren. Falls die Einträge von P_2S hinreichend symmetrisch sind und die Diagonale nahezu nullfrei ist, wird die 2×2-Strategie verwendet. Sonst wird die unsymmetrische Strategie benutzt.

- *unsymmetrisch*: Die Spaltenvorsortierung von S wird mit Hilfe einer modifizierten Version von COLAMD [DGLN04] berechnet. Der Algorithmus findet eine symmetrische Permutation Q von der Matrix S^TS (ohne S^TS direkt zu berechnen). Dies ist eine gute Wahl für Q, da die Cholesky Faktoren von $(SQ)^T(SQ)$ bei der unsymmetrischen LU-Zerlegung ($PSQ = LU$) eine obere Schranke für die Faktoren von U bilden. Diese Schranke hängt nicht von der Wahl von P ab. Die modifizierte Version von COLAMD berechnet ebenfalls den Eliminationsbaum und sortiert ihn mittels post-order. Während der Faktorisierung kann die Zeilenvorsortierung modifiziert werden. Spalten innerhalb einer Superspalte können umsortiert werden, um Fill-in zu reduzieren. Es wird teilweise Threshold-Pivotisierung ohne Berücksichtigung der Diagonaleinträge verwendet. Innerhalb einer gegebenen Zeile j kann jedes Element a_{ij} gewählt werden, falls $|a_{ij}| \geq 0.1 \max |a_{*j}|$.

- *symmetrisch*: Die Spaltenvorsortierung wird von AMD [ADD96] durch die Anwendung auf $S + S^T$ berechnet, gefolgt von einer Sortierung des supernodalen Eliminationsbaumes von $S + S^T$ mittels pre-order. An der Spaltenvorsortierung wird während der numerischen Faktorisierung keine Veränderung vorgenommen. Dabei wird teilweise Threshold-Pivotisierung verwendet, wobei die Diagonale stark gewichtet wird. Das Diagonalelement wird gewählt, falls $|a_{jj}| \geq 0.001 \max |a_{*j}|$. Ansonsten werden dünnbesetzte Zeilen nach dem gleichen Kriterium ausgewählt wie im unsymmetrischen Fall (finde die dünnbesetzteste Pivot-Zeile unter Verwendung eines Threshold von 0.1).

- 2×2: Die symmetrische Startegie wird auf die Matrix P_2S angewendet, anstatt auf S.

Nachdem die Strategie ausgewählt worden ist, wird die Faktorisierung der Matrix A aufgeschlüsselt in die Zerlegung einer Folge von dichtbesetzten Frontalmatrizen. Diese Matrizen sind über einen supernodalen Spaltenbaum miteinander verbunden, indem jeder Knoten des Baumes eine Frontalmatrix darstellt. In der Analysephase werden auch obere Schranken für den Speicherbedarf, die Anzahl der Fließkomma-Operationen und die Anzahl der Nicht-Null-Einträge der LU-Zerlegung ermittelt.

UMFPACK faktorisiert jede Kette aus Frontalmatrizen in einer einzigen Arbeitsmatrix. Eine Kette aus Frontalmatrizen ist dabei eine Folge von Fron-

ten, bei denen i Eltern von $i+1$ im supernodalen Spalteneliminationsbaum sind. UMFPACK ist eine rechtsseitige Methode, die auf äußeren Produkten basiert. Zum k-ten Schritt der Gaußelimination erfolgt das Update der Untermatrix A_k in Form einer impliziten Summierung einer Menge von dichten Untermatrizen, die auftreten, wenn die Frontalmatrizen faktorisiert werden und ihre Pivotzeilen und Pivotspalten eliminiert werden.

Jede Frontalmatrix repräsentiert die Elimination einer oder mehrerer Spalten. Jede Spalte von A wird in einer speziellen Frontalmatrix eliminiert. Welche Spalte dabei von welcher Frontalmatrix eliminiert wird, bestimmt man in der Voranalysephase. In dieser Phase wird auch die "worst-case"-Größe jeder Frontalmatrix bestimmt, indem jede beliebige Zeile bzw. Spalte als Pivotkandidat angenommen wird. Die Menge an Kandidaten von Pivotspalten einer einzelnen Frontalmatrix bilden eine einzelne "Superspalte". Aus der Sicht der Analysephase ist jeder Kandidat für eine Pivotspalte in der Frontalmatrix identisch (im Hinblick auf nicht-null Einträge). Gleiches gilt für jede Zeile. Existierende linksseitige Methoden zur numerischen Faktorisierung haben keine weiteren Informationen. In Hinblick auf die Reduzierung des Fill-in können sie nicht zwischen Kanditaten für Pivotspalten innerhalb einer Superspalte unterscheiden, genauso wenig wie zwischen Kandidaten für Pivotzeilen.

Im Gegensatz hierzu hat die rechtsseitige numerische Faktorisierungsphase von UMFPACK mehr Informationen als die Analysisphase. Es benutzt diese Informationen, um die Spalten jeder Frontalmatrix umzuordnen, so dass der Fill-in reduziert wird. UMFPACK vertauscht nur die Spalten in einer Superspalte, so dass keine Änderungen an den oberen Schranken aus der symbolischen bzw. der Analysis-Phase erfolgen. Da die Anzahl der Nicht-Null-Einträge in jeder Zeile und Spalte beibehalten werden (genauer, COLMMD-style approximate degrees [GMS92]), kann eine Pivotzeile sowohl auf dem Kriterium des Erhaltens der Dünnbesetztheit als auch nach numerischen Gesichtspunkten (relaxed threshold partial pivoting) gewählt werden.

Die numerische Faktorisierung verfeinert die Spaltenumordnung Q, indem die Pivotspalten innerhalb einer Front umgeordnet werden, und sie berechnet die Zeilenumordnung P, welche die duale Rolle von der Reduzierung des Fill-in und der Erhaltung der numerischen Genauigkeit hat.

4.1.2 GMRES (Generalized Minimal RESidual) -Verfahren

Das GMRES-Verfahren (vgl. z.B. [Gr97]) ist ein Krylov-Raum-Verfahren zur Lösung linearer Gleichungssysteme $Ax = b$. Der k-te zur Matrix A und Vektor b gehörige Krylov-Raum wird definiert durch:

$$W_k := span\{b, Ab, \ldots, A^{k-1}b\}, \quad k = 1, 2, \ldots .$$

Beim GMRES-Verfahren wird der Raum

$$W_k := span\{r_0, Ar_0, \ldots, A^{k-1}r_0\},$$

mit $r_0 = b - Ax_0$, x_0 Startvektor, eingeführt. Gesucht werden Iterierte $x_k \in x_0 + W_k$, für die die 2-Norm des Residuums minimal ist. Zur Konstruktion einer Orthonormalbasis von W_k wird die sogenannte *Arnoldi-Methode* verwendet.

Arnoldi-Verfahren:

(i) Wähle Startvektor v_1 mit $||v_1|| = 1$.

(ii) Für $j = 1, 2, \ldots$
$\tilde{v}_{j+1} = Av_j$;
Für $i = 1, \ldots, j$
$h_{i,j} = (\tilde{v}_{j+1}, v_i)$;
$\tilde{v}_{j+1} = \tilde{v}_{j+1} - h_{i,j}v_i$;
$h_{j+1,j} = ||\tilde{v}_{j+1}||$;
$v_{j+1} = \tilde{v}_{j+1}/h_{j+1,j}$;

Zu Beginn des GMRES-Verfahrens setzt man $v_1 = r_0/||r_0||$. Bezeichnet man mit V_k die $n \times k$-Matrix, die aus den Spaltenvektoren $v_1, \ldots, v_k$ besteht, kann man das Arnoldi-Verfahren auch in Matrix-Form schreiben als:

$$AV_k = V_k H_k + h_{k+1,k} v_k \xi_k^T = V_{k+1} H_{k+1,k}, \tag{4.2}$$

mit der oberen Hessenberg-Matrix $H_k = (h_{i,j})_{\substack{j = 1, \ldots, k \\ i = 1, \ldots, \min(j+1, k)}} \in \mathbb{R}^{k,k}$, $\xi_k \in \mathbb{R}^k$ k-ter Einheitsvektor. Ferner ist

$$H_{k+1,k} = \begin{pmatrix} H_k \\ 0 \dots 0 \; h_{k+1,k} \end{pmatrix}.$$

Die Iterierte x_k im k-ten Schritt ergibt sich beim GMRES-Verfahren nun wegen $x_k \in x_0 + W_k$ zu

$$x_k = x_0 + V_k y_k,$$

wobei y_k noch zu bestimmen ist. Ziel ist es dabei, das Residuum zu minimieren, d.h.

$$\begin{aligned} r_k &= b - Ax_k = b - (Ax_0 + AV_k y_k) \\ &= r_0 - AV_k y_k \end{aligned}$$

soll in der 2-Norm minimal werden. Somit muss also der Vektor y_k das folgende Least-Squares-Problem lösen:

$$\begin{aligned} \min_y ||r_0 - AV_k y|| &= \min_y ||r_0 - V_{k+1} H_{k+1,k} y|| \\ &= \min_y ||V_{k+1}(\beta\xi_1 - H_{k+1,k} y)|| \\ &= \min_y ||\beta\xi_1 - H_{k+1,k} y||, \end{aligned} \tag{4.3}$$

wobei $\beta = ||r_0||, \xi_1$ der erste Einheitsvektor im $\mathbb{R}^{k+1}$ ist.

Zu lösen ist nun also noch das Least-Squares-Problem (4.3). Dabei wird $H_{k+1,k}$ mit Hilfe der QR-Zerlegung in eine unitäre Matrix $F \in \mathbb{C}^{k+1,k+1}$ (d.h. $F^H_{i,j} = F_{j,i}$) und eine obere Dreiecksmatrix $R \in \mathbb{R}^{k+1,k}$ zerlegt mit

$$R = \begin{pmatrix} R_{k\times k} \\ 0 \dots 0 \end{pmatrix},$$

($R_{k\times k} \in \mathbb{R}^{k,k}$ obere Dreiecksmatrix). y_k erhält man, indem man die Lösung des Gleichungssystems

$$R_{k\times k} y = \beta (F\xi_1)_{k\times 1} \tag{4.4}$$

bestimmt. $(F\xi_1)_{k\times 1}$ bezeichnet die ersten k Einträge der ersten Spalte von F.

Zur Verminderung des Aufwands kann man aus der QR-Zerlegung von $H_{k+1,k}$ auf die QR-Zerlegung der Matrix $H_{k+2,k+1}$ schließen. Dabei nutzt man aus, dass man die QR-Zerlegung der Matrix $H_{k+1,k}$ mittels *Givens-Rotationen* erhalten kann (vgl. [Gr97]). Insgesamt ergibt sich dann das

GMRES-Verfahren:

(i) Wähle einen Startvektor x_0, berechne $r_0 = b - AX_0$ und setze $v_1 = r_0/||r_0||$.

(ii) Für $k = 1, 2, \ldots$

Führe das Arnoldi-Verfahren durch und berechne damit
v_{k+1} und $h_{i,j},\ i = 1, \ldots, k+1$
$x_k = x_0 + V_k y_k$,
wobei y_k die Lösung des Least-Squares Problem (4.3) ist unter der Verwendung der Givens-Rotationen.

(iii) Sobald die gewünschte Genauigkeit erreicht ist, brich das Verfahren ab.

4.2 Postprocessing

Wie bereits in Kap. 1.3 erklärt, erhält man eine stabile Diskretisierung mit zentralen Differenzen nur, falls die Schrittweitenbeschränkung $h \leq C \cdot \varepsilon$ erfüllt ist. Da dies zu viel zu großen Gleichungssystemen führen würde, diskretisiert man die Konvektions-Diffusions-Gleichung mittels einseitiger Differenzen, dem sogenannten Upwind-Verfahren. Nachdem das adaptive Verfeinerungsverfahren konvergiert ist, hat sich das Schrittweitenverhältnis von h und ε im Bereich des Layers stark verändert.

In den meisten Fällen gibt es nun dort Gebiete, für die die Schrittweitenbeschränkung $h \leq C \cdot \varepsilon$ erfüllt ist. Da gerade der Bereich um den Layer in der Regel mit den größten Approximationsfehlern behaftet ist, versucht man nun, diesen mit höherer Genauigkeit aufzulösen. Dazu betrachtet man die Indexmenge I, welche alle Punkte des diskreten Gitters Ω_h umfasst, d.h.

$\Omega_h = \{(x_i, y_j) \ : \ i, j \in I\}$. Hierfür definiert man zwei Teilmengen $I^0, I^1 \subseteq I$ mit

$$I^0 := \left\{ i \in I \ : \ \sum_{\substack{i \neq j \\ j \in I}} |L_h^0(i,j)| \leq |L_h^0(i,i)| \right\}, \tag{4.5}$$

$$I^1 := I \setminus I^0. \tag{4.6}$$

Dabei ist L_h^0 der zentrale Differenzen-Operator für die Konvektions-Diffusions-Gleichung. I^0 stellt dabei die Menge aller Punkte dar, für die eine Diskretisierung mit zentralen Differenzen zu einer M-Matrix führen würde (vgl. Kap. (1.2)). Verwendet man nun als Diskretisierungsoperator eine Kombination aus zentralen und einseitigen Differenzen, d.h. zentrale Differenzen für alle Punkte in I^0 und upwind Differenzen für alle Punkte in I^1, so erhält man eine stabile Diskretisierung in Form einer M-Matrix. Der so zusammengesetzte Operator erzielt punktweise im Bereich des Layers die Ordnung 2. Da sich die Lösung in diesem Bereich besonders stark ändert und außerhalb häufig sehr glatt ist, hat man auf diese Weise mit einer deutlichen Verbesserung der Genauigkeit der diskreten Lösung zu rechnen.

Kapitel 5

Numerische Ergebnisse

Im folgenden Kapitel soll das entwickelte adaptive Upwind-Verfahren an verschiedenen Beispielen getestet werden. Auf dem adaptiv verfeinerten Gitter wird jeweils das entwickelte Postprocessing angewendet und die erzielte Verbesserung diskutiert.

Die folgenden Probleme werden für

$$-\varepsilon\Delta u + b\cdot\nabla u + cu = f, \quad x\in\Omega=(0,1)^2, \tag{5.1}$$

betrachtet. Der Fehler wird in den diskreten Normen

$$\begin{aligned} ||u_h||_{L_1} &= \sum_{S\in\$} h_S^2 \sum_{p\in S} |u_h(p)-u(p)| \\ ||u_h||_{L_\infty} &= \sup_{p\in\Omega_h} |u_h(p)-u(p)| \end{aligned}$$

angegeben. Dabei bezeichnet \$ die Menge aller Quadrate in Ω_h und h_S die Kantenlänge eines Quadrates.

5.1 Beispiel 1

Das erste Beispiel behandelt den Fall eines Randlayers. Die Parameter hierfür sind $\varepsilon = 10^{-3}$, $b\equiv(1,1)$, $c\equiv 1$ und die rechte Seite $f(x)$ derart, dass die exakte Lösung $u(x,y)=\exp\left(-\frac{(x-1)(y-1)}{\varepsilon}\right)$ ist. Die Lösung u ist nahezu

Tabelle 5.1: Beispiel 1

$\varepsilon = 10^{-3}$	$\delta = 10^{-2}$		$\delta = 10^{-3}$	
#Iteration	8		9	
dof	53729		406833	
	Finalgrid	Postprocessing	Finalgrid	Postprocessing
$\|\|u - u_h\|\|_{L_1}$	$4.512 \cdot 10^{-3}$	$1.384 \cdot 10^{-3}$	$1.536 \cdot 10^{-3}$	$1.646 \cdot 10^{-4}$
$\frac{\|\|u-u_h\|\|_{L_1}}{\|\|u\|\|_{L_1}}$	$1.497 \cdot 10^{-1}$	$4.595 \cdot 10^{-2}$	$5.128 \cdot 10^{-2}$	$5.495 \cdot 10^{-3}$
$\|\|u - u_h\|\|_{L_\infty}$	$7.789 \cdot 10^{-2}$	$2.305 \cdot 10^{-2}$	$2.782 \cdot 10^{-2}$	$1.920 \cdot 10^{-3}$

konstant ($u \equiv 0$) auf fast gesamt Ω. Nur am rechten und oberen Rand ($x = 1,$ bzw $y = 1$) steigt die Lösung sprunghaft auf 1 an (vgl. Abb. (5.1)).

Tabelle 5.1 zeigt die numerischen Ergebnisse für Beispiel 1. Man erkennt deutlich, dass die Güte der Lösung sowie der Rechenaufwand stark von der Fehlerschätzerschranke δ abhängen. Da es sich bei dem hier verwendeten Fehlerschätzer nicht um einen Schätzer im eigentlichen Sinne, sondern mehr um einen Fehlerindikator handelt, bleibt die Wahl von δ heuristisch. Messungen haben ergeben, dass in der Regel $\delta = \varepsilon$ eine gute Wahl ist.

Die Abbildungen (5.2) und (5.3) zeigen das adaptiv verfeinerte Gitter. Dabei ist deutlich zu erkennen, dass das Gitter ausschließlich in Richtung des Randlayers verfeinert worden ist. Das Verfahren erfasst den Verlauf des Layers gut und erreicht das höchste Verfeinerungslevel im Bereich um den Layer herum.

Einen Plot des Fehlers $|u - u_h|$ zeigen die Abbildungen (5.4) und (5.5). Wie zu erkennen ist, konnte der Fehler durch das Postprocessing deutlich reduziert werden. Tabelle (5.1) ist zu entnehmen, dass sich der Fehler für $\delta = 10^{-3}$ um eine Zehnerpotenz verkleinert hat. Für $\delta = 10^{-2}$ verringert sich der Fehler durch das Postprocessing nicht so deutlich. Das liegt daran, dass durch die gröbere Wahl von δ nicht so weit verfeinert wurde. Folglich ist der Bereich der Punkte kleiner, in denen ein Postprocessing überhaupt erlaubt ist. Dementsprechend fällt die Fehlerreduzierung hierbei geringer aus.

Tabelle 5.2: Iterationsschritte Beispiel 1

	$\delta = 10^{-2}$		$\delta = 10^{-3}$		äquidistant	
Level	dof	$\|\|u - u_h\|\|_{L_\infty}$	dof	$\|\|u - u_h\|\|_{L_\infty}$	dof	$\|\|u - u_h\|\|_{L_\infty}$
0	289	$8.166 \cdot 10^{-2}$	289	$8.166 \cdot 10^{-2}$		
1	445	$2.840 \cdot 10^{-1}$	528	$2.840 \cdot 10^{-1}$		
2	1019	$2.160 \cdot 10^{-1}$	1036	$2.104 \cdot 10^{-1}$	1089	$2.840 \cdot 10^{-1}$
3	2086	$1.958 \cdot 10^{-1}$	2444	$1.947 \cdot 10^{-1}$		
4	4540	$1.951 \cdot 10^{-1}$	5834	$1.904 \cdot 10^{-1}$	4225	$2.104 \cdot 10^{-1}$
5	10809	$1.952 \cdot 10^{-1}$	15434	$1.904 \cdot 10^{-1}$	16641	$1.946 \cdot 10^{-1}$
6	23449	$1.286 \cdot 10^{-1}$	42794	$1.271 \cdot 10^{-1}$		
7	42661	$8.163 \cdot 10^{-2}$	116405	$7.410 \cdot 10^{-2}$	66049	$1.904 \cdot 10^{-1}$
8	53729	$7.789 \cdot 10^{-2}$	246577	$4.199 \cdot 10^{-2}$	263169	$1.904 \cdot 10^{-1}$
9			406883	$2.782 \cdot 10^{-2}$	491401	$1.693 \cdot 10^{-1}$
pp	53729	$2.305 \cdot 10^{-2}$	406833	$1.920 \cdot 10^{-3}$	491401	$2.917 \cdot 10^{-2}$

Hält man sich aber vor Augen, dass der Aufwand des Postprocessings vernachlässigbar gering ist, so lohnt es sich auch für gröbere Schranken δ.

Tabelle (5.2) stellt den Verlauf des Fehlers $||u - u_h||_{L_\infty}$ dar und vergleicht die Ergebnisse mit dem Fall des äquidistanten Gitters. Der Fehler nach dem Postprocessing (pp) ist in der letzten Zeile aufgeführt. Es lässt sich die typische Stagnation des Fehlers nach den ersten Verfeinerungsschritten beobachten. Dies liegt daran, dass der Bereich außerhalb des Layers bereits nach einigen Schritten gut aufgelöst wird. Bis die Gitterschrittweite h klein genug ist, um auch den Layer zu erfassen, scheint der Fehler zu stagnieren.

Betrachtet man den Fall des äquidistanten Gitters, so lässt sich deutlich der Vorteil des adaptiven Verfahrens erkennen. Es werden im äquidistanten

Fall ca. 500000 Gitterpunkte benötigt, um einen Fehler in der Größenordnung von $2 \cdot 10^{-2}$ zu erhalten. Das adaptive Verfahren benötigt dazu nur knapp ein Zehntel der Punkte (ca 50000).

5.2 Beispiel 2

Das zweite Beispiel behandelt den Fall eines Eckenlayers. Die Parameterwerte hierbei sind $\varepsilon = 10^{-4},\ b = [1/4-x, 1/4-y],\ c \equiv 1$. Die rechte Seite $f(x)$ wurde so gewählt, dass die exakte Lösung $u(x,y) = \exp\left(-\frac{x^6+y^6}{\varepsilon}\right)$ ist. Zu bemerken ist, dass $b(x,y)$ in Ω das Vorzeichen wechselt. Es entstehen also Gebiete, die unterschiedlich diskretisiert werden müssen.

Auch in diesem Fall ist die Lösung in großen Teilen des Gebietes nahezu konstant ($u \equiv 0$) und steigt in der rechten oberen Ecke fast sprunghaft an ($u \approx 1$) (vgl. Abbildung (5.6)).

Tabelle (5.3) zeigt die numerischen Ergebnisse für dieses Beispiel. Es wurden Rechnungen für verschiedene Schranken δ durchgeführt. Hierbei lässt sich gut erkennen, dass man δ in Abhängigkeit von ε wählen muss. Eine zu grobe Wahl von δ liefert zum einen in der Regel nicht die gewünschte Genauigkeit. Zum anderen ist eine Anwendung des Postprocessings hierfür nicht sinnvoll, da aufgrund der recht groben Verfeinerung keine Bereiche existieren, in denen man mit zentralen Differenzen diskretisieren kann.

Die Abbildungen (5.7) und (5.8) zeigen das adaptiv verfeinerte Gitter für $\delta = 10^{-4}$. Man kann gut erkennen, dass das Verfahren stets in Richtung des Layers verfeinert hat und den Bereich um diesen gut aufgelöst hat.

Einen Plot des Fehlers $|u - u_h|$ zeigen die Abbildungen (5.9) und (5.10) vor bzw. nach dem Postprocessing. Auch hier lässt sich eine deutliche Reduzierung des Fehlers erkennen.

Der Verlauf der Adaption wird in Tabelle (5.4) dargestellt. Wie schon im ersten Beispiel erzielt dabei das adaptive Verfahren deutlich bessere Ergebnisse als die äquidistante Verfeinerung. Um einen Fehler in der Größenordnung von 10^{-3} zu bekommen, werden weniger als ein Zehntel an Gitterpunkten benötigt.

Tabelle 5.3: Beispiel 2

$\varepsilon = 10^{-4}$	$\delta = 10^{-2}$		$\delta = 10^{-3}$	
#Iteration	5		5	
dof	1096		22947	
	Finalgrid	Postprocessing	Finalgrid	Postprocessing
$\|\|u-u_h\|\|_{L_1}$	$2.699 \cdot 10^{-3}$	$2.703 \cdot 10^{-3}$	$7.349 \cdot 10^{-4}$	$6.204 \cdot 10^{-4}$
$\frac{\|\|u-u_h\|\|_{L_1}}{\|\|u\|\|_{L_1}}$	$1.677 \cdot 10^{-2}$	$1.679 \cdot 10^{-2}$	$4.596 \cdot 10^{-3}$	$3.880 \cdot 10^{-3}$
$\|\|u-u_h\|\|_{L_\infty}$	$2.768 \cdot 10^{-2}$	$2.768 \cdot 10^{-2}$	$6.586 \cdot 10^{-3}$	$6.268 \cdot 10^{-3}$
	$\delta = 5 \cdot 10^{-4}$		$\delta = 10^{-4}$	
#Iteration	7		8	
dof	52959		259282	
	Finalgrid	Postprocessing	Finalgrid	Postprocessing
$\|\|u-u_h\|\|_{L_1}$	$4.725 \cdot 10^{-4}$	$3.328 \cdot 10^{-4}$	$2.016 \cdot 10^{-4}$	$7.129 \cdot 10^{-5}$
$\frac{\|\|u-u_h\|\|_{L_1}}{\|\|u\|\|_{L_1}}$	$2.956 \cdot 10^{-3}$	$2.082 \cdot 10^{-3}$	$1.236 \cdot 10^{-3}$	$4.462 \cdot 10^{-4}$
$\|\|u-u_h\|\|_{L_\infty}$	$4.721 \cdot 10^{-3}$	$2.895 \cdot 10^{-3}$	$1.815 \cdot 10^{-3}$	$9.478 \cdot 10^{-4}$

5.3 Beispiel 3

In Beispiel 3 wird der Fall eines kreisrunden inneren Layers (Hutlayer) behandelt. Die Parameter hierfür sind $\varepsilon = 5 \cdot 10^{-3}$,
$b(x,y) = ((4x-2)u^e(x,y), (4y-2)u^e(x,y))\,,\ c(x,y) = 4u^e(x,y),\ f(x,y) = 4.$
Die exakte Lösung ist

$$u^e(x,y) = \tanh\left[\frac{1}{\varepsilon}\left(\frac{1}{16} - \left(x-\frac{1}{2}\right)^2 - \left(y-\frac{1}{2}\right)^2\right)\right].$$

Abbildung (5.11) zeigt einen Plot der exakten Lösung. Der Graph ist auf fast ganz Ω konstant ($u \equiv 0$ oder $u \equiv 1$) und ändert seinen Funktionswert entlang eines kreisrunden Ringes im Inneren des Gebietes. Diesen Hutlayer zu finden und gut aufzulösen, stellt für viele adaptive Verfahren ein Problem dar. Wie

Tabelle 5.4: Iterationsschritte Bsp. 2

	$\delta = 10^{-3}$		$\delta = 10^{-4}$		äquidistant	
Level	dof	$\|\|u - u_h\|\|_{L_\infty}$	dof	$\|\|u - u_h\|\|_{L_\infty}$	dof	$\|\|u - u_h\|\|_{L_\infty}$
0	289	$1.861 \cdot 10^{-1}$	289	$1.861 \cdot 10^{-1}$		
1	384	$1.136 \cdot 10^{-1}$	410	$1.136 \cdot 10^{-1}$		
2	728	$6.511 \cdot 10^{-2}$	783	$6.511 \cdot 10^{-2}$	1089	$1.136 \cdot 10^{-1}$
3	4234	$1.873 \cdot 10^{-2}$	2027	$3.651 \cdot 10^{-2}$		
4	12008	$1.013 \cdot 10^{-2}$	6374	$1.849 \cdot 10^{-2}$	4225	$6.511 \cdot 10^{-2}$
5	22947	$6.586 \cdot 10^{-3}$	20975	$9.434 \cdot 10^{-3}$	16641	$3.561 \cdot 10^{-2}$
6			61706	$4.821 \cdot 10^{-3}$	66049	$1.848 \cdot 10^{-2}$
7			189465	$2.571 \cdot 10^{-3}$		
8			259282	$1.815 \cdot 10^{-3}$	263169	$9.405 \cdot 10^{-3}$
pp	22947	$6.268 \cdot 10^{-3}$	259282	$9.478 \cdot 10^{-4}$	263169	$8.011 \cdot 10^{-3}$

Abbildung (5.12) und (5.13) zeigen, löst unser Verfahren diese Aufgabe sehr gut.

Die numerischen Ergebnisse sind in Tabelle (5.5) dargestellt. Auch hier wird die Abhängigkeit zwischen ε und δ deutlich. Je nach benötigter Genauigkeit der Lösung empfiehlt es sich auch hierbei wieder, $\delta \approx \varepsilon$ zu wählen.

Bemerkenswert bei diesem Beispiel ist, dass das Postprocessing kaum Verbesserung erzielt. Dies liegt daran, dass beim Beispiel des Hutlayers die Punkte, an denen eine zentrale Diskretisierung möglich ist, weit verstreut liegen. Bei den anderen beiden Beispielen zuvor lagen diese Punkte dicht und zusammenhängend im Bereich des Layers. Offensichtlich führen die vereinzelt liegenden, zentral diskretisierten Punkte kaum zu einer Verbesserung der Lösung. Eine feinere Auflösung des Gitters würde sicherlich auch für diesen Fall eine deutlichere Fehlerreduzierung durch das Postprosessing erzielen.

Tabelle 5.5: Beispiel 3

$\varepsilon = 5 \cdot 10^{-3}$	$\delta = 10^{-1}$		$\delta = 10^{-2}$	
#Iteration	6		6	
dof	2413		10145	
	Finalgrid	Postprocessing	Finalgrid	Postprocessing
$\|\|u - u_h\|\|_{L_1}$	$3.961 \cdot 10^{-2}$	$4.132 \cdot 10^{-2}$	$1.528 \cdot 10^{-2}$	$1.453 \cdot 10^{-2}$
$\frac{\|\|u-u_h\|\|_{L_1}}{\|\|u\|\|_{L_1}}$	$1.012 \cdot 10^{-2}$	$1.056 \cdot 10^{-2}$	$3.587 \cdot 10^{-3}$	$3.667 \cdot 10^{-3}$
$\|\|u - u_h\|\|_{L_\infty}$	$1.489 \cdot 10^{-1}$	$1.489 \cdot 10^{-1}$	$7.030 \cdot 10^{-2}$	$7.021 \cdot 10^{-2}$
	$\delta = 10^{-3}$		$\delta = 10^{-4}$	
#Iteration	6		7	
dof	79157		216205	
	Finalgrid	Postprocessing	Finalgrid	Postprocessing
$\|\|u - u_h\|\|_{L_1}$	$5.078 \cdot 10^{-3}$	$4.304 \cdot 10^{-3}$	$3.739 \cdot 10^{-3}$	$3.072 \cdot 10^{-3}$
$\frac{\|\|u-u_h\|\|_{L_1}}{\|\|u\|\|_{L_1}}$	$1.297 \cdot 10^{-3}$	$1.100 \cdot 10^{-3}$	$9.555 \cdot 10^{-4}$	$7.851 \cdot 10^{-4}$
$\|\|u - u_h\|\|_{L_\infty}$	$2.465 \cdot 10^{-2}$	$2.217 \cdot 10^{-2}$	$1.627 \cdot 10^{-2}$	$9.397 \cdot 10^{-3}$

Den Vorteil der adaptiven Verfeinerung gegenüber der äqudistanten Verfeinerung verdeutlicht Tabelle (5.6). Um einen Fehler in der Größenordnung von $2 \cdot 10^{-2}$ zu bekommen, wird bei dem adaptiven Verfahren nur knapp ein Viertel an Gitterpunkten benötigt.

Tabelle 5.6: Iterationsschritte Bsp. 3

	$\delta = 10^{-3}$		$\delta = 10^{-4}$		äquidistant	
Level	dof	$\|\|u - u_h\|\|_{L_\infty}$	dof	$\|\|u - u_h\|\|_{L_\infty}$	dof	$\|\|u - u_h\|\|_{L_\infty}$
0	289	$5.057 \cdot 10^{-1}$	289	$5.057 \cdot 10^{-1}$		
1	733	$2.272 \cdot 10^{-1}$	773	$2.272 \cdot 10^{-1}$		
2	1853	$1.812 \cdot 10^{-1}$	1945	$1.812 \cdot 10^{-1}$	1089	$1.136 \cdot 10^{-1}$
3	4701	$1.206 \cdot 10^{-1}$	4989	$1.206 \cdot 10^{-1}$	4225	$1.812 \cdot 10^{-1}$
4	12025	$7.319 \cdot 10^{-2}$	13149	$7.315 \cdot 10^{-2}$	16641	$1.205 \cdot 10^{-1}$
5	31229	$4.166 \cdot 10^{-2}$	35169	$4.125 \cdot 10^{-2}$		
6	79157	$2.465 \cdot 10^{-2}$	94269	$2.302 \cdot 10^{-2}$	66049	$7.307 \cdot 10^{-2}$
7			216205	$1.627 \cdot 10^{-2}$	263169	$4.090 \cdot 10^{-2}$
pp	79157	$2.217 \cdot 10^{-2}$	216205	$9.397 \cdot 10^{-3}$	263169	$2.743 \cdot 10^{-2}$

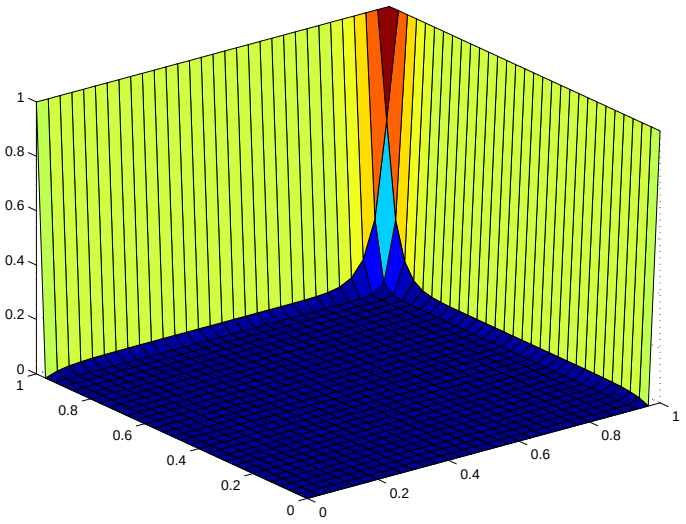


Abbildung 5.1: Exakte Lösung für Beispiel 1

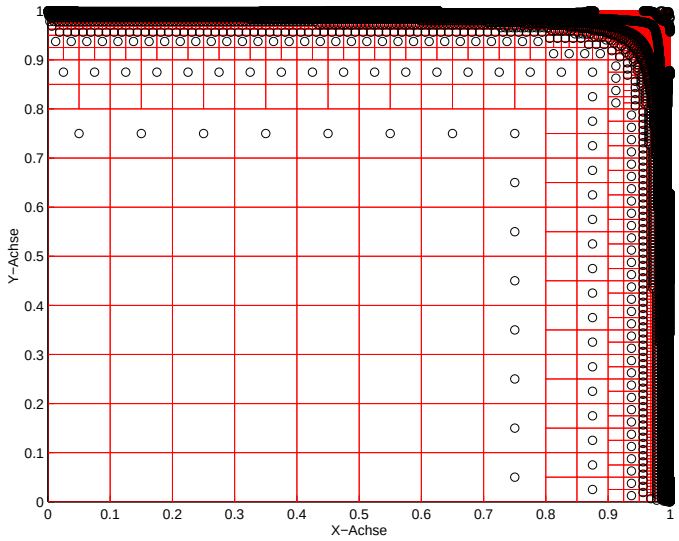


Abbildung 5.2: Adaptives Gitter für Beispiel 1

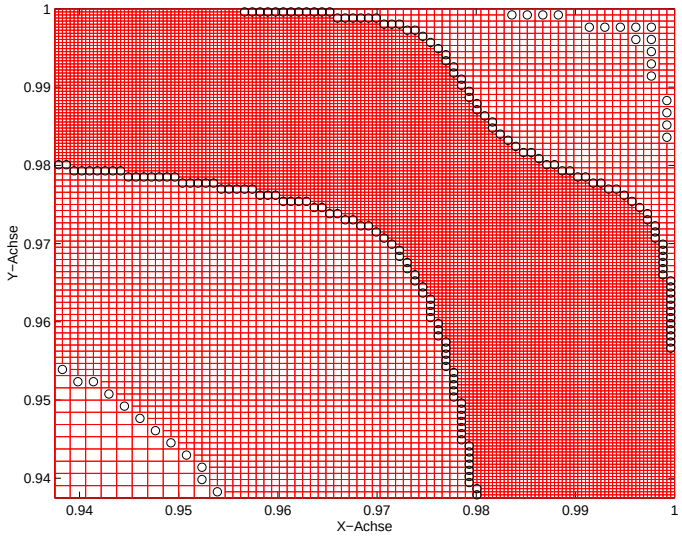


Abbildung 5.3: Zoom auf das Gitter für Beispiel 1

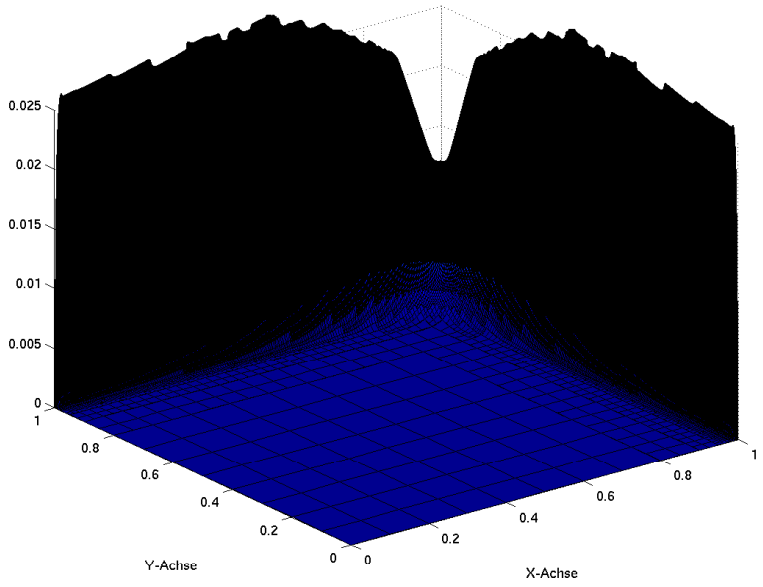


Abbildung 5.4: Fehler $|u - u_h|$ für Beispiel 1

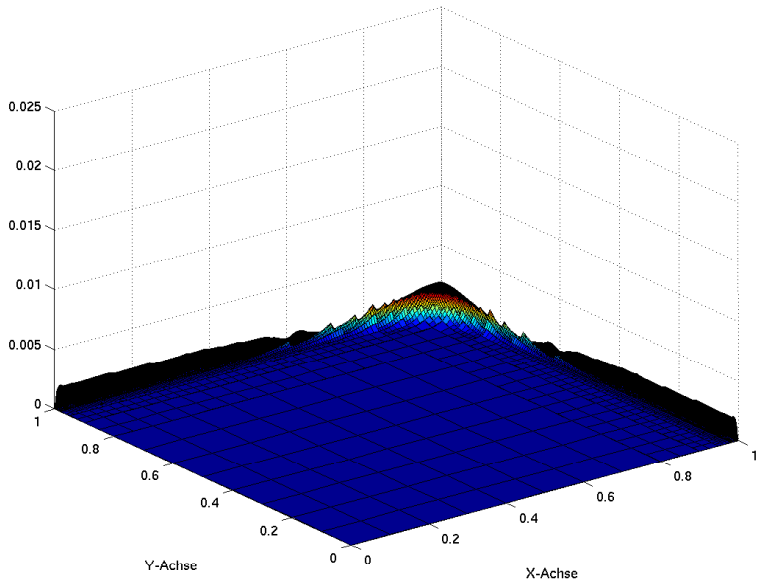


Abbildung 5.5: Fehler nach Postprocessing $|u - u_h|$ für Beispiel 1

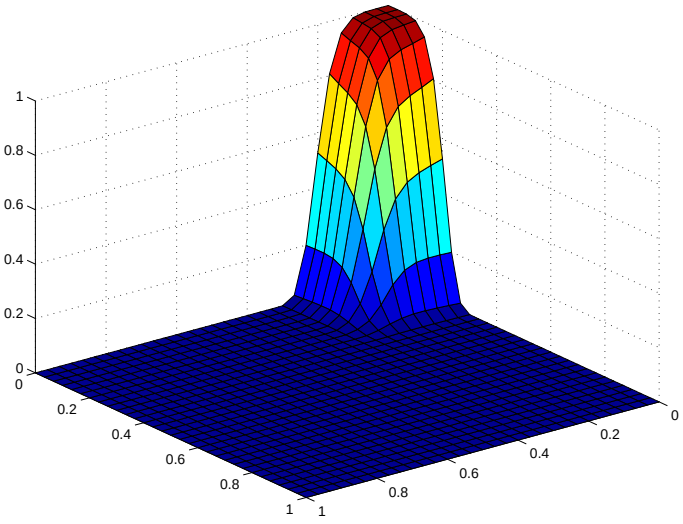


Abbildung 5.6: Exakte Lösung für Beispiel 2

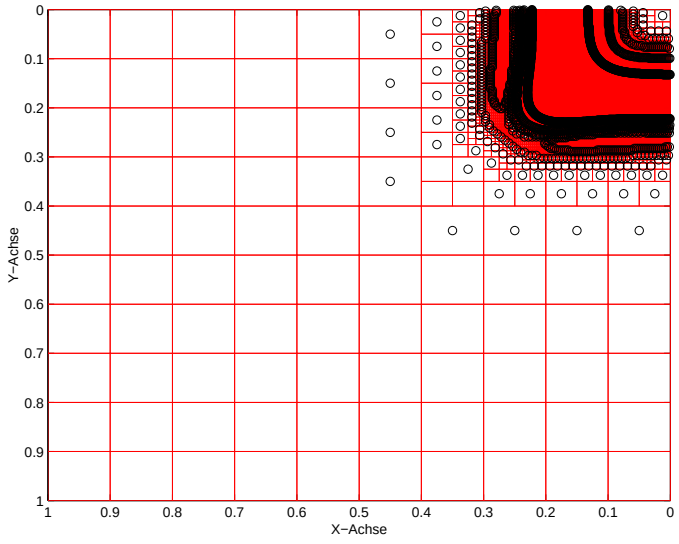


Abbildung 5.7: Adaptives Gitter für Beispiel 2

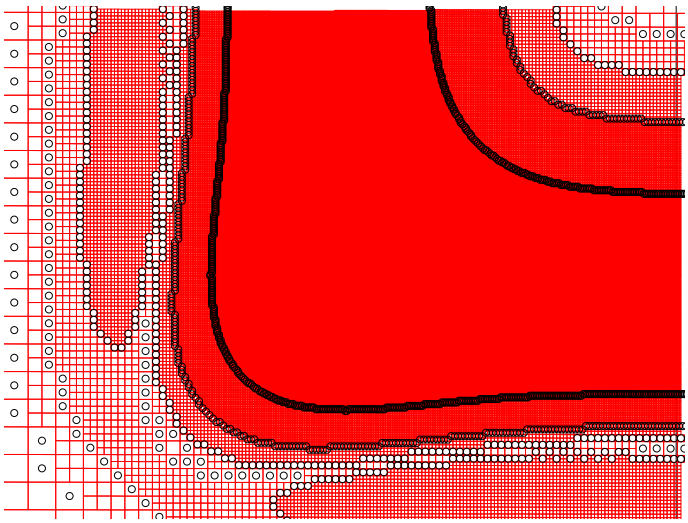


Abbildung 5.8: Zoom auf das Gitter für Beispiel 2

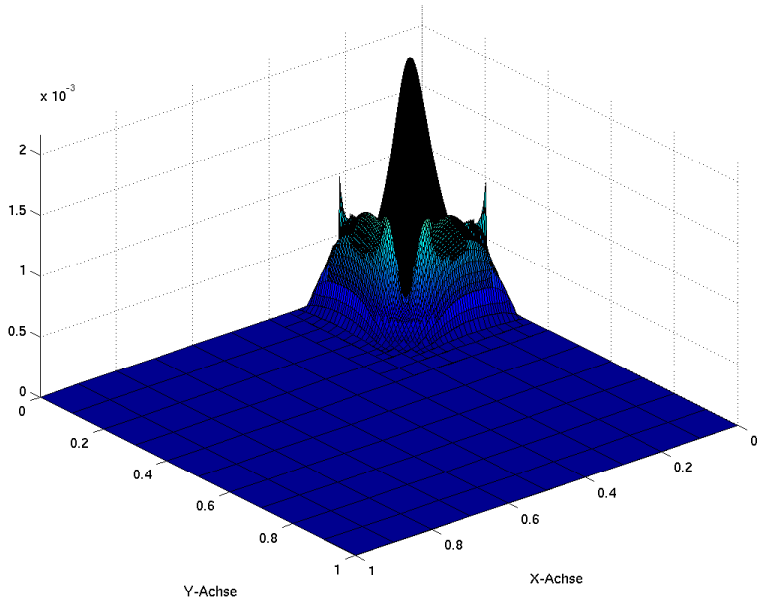


Abbildung 5.9: Fehler $|u - u_h|$ für Beispiel 2

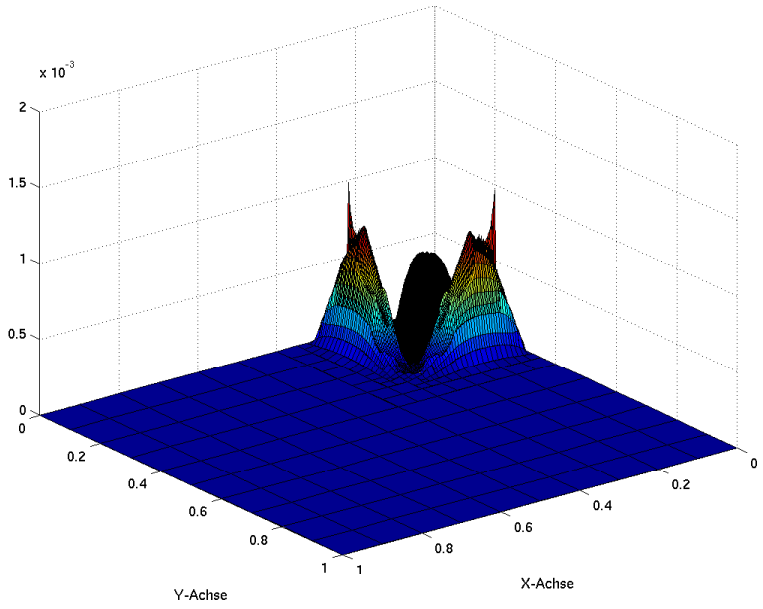


Abbildung 5.10: Fehler nach Postprocessing $|u - u_h|$ für Beispiel 2

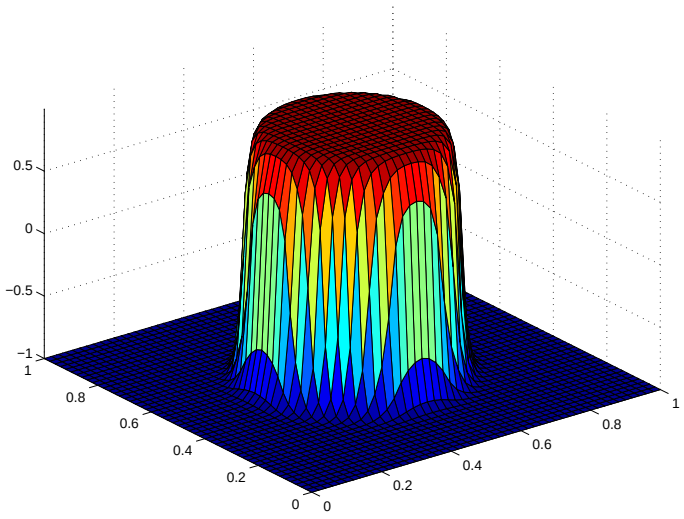


Abbildung 5.11: Exakte Lösung für Beispiel 3

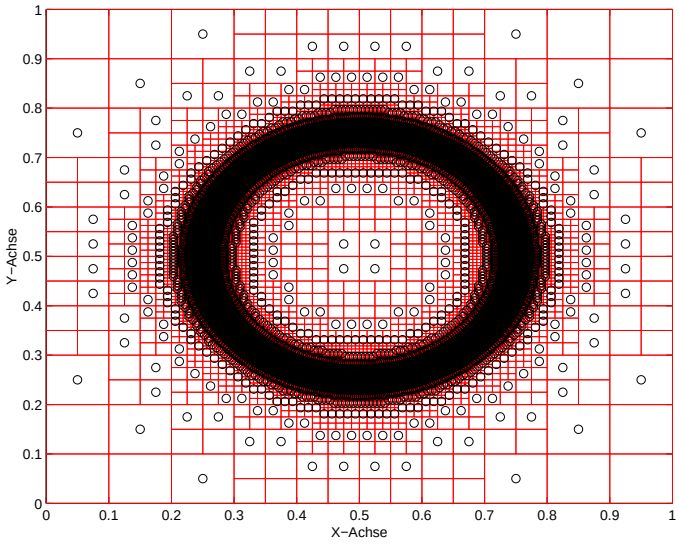


Abbildung 5.12: Adaptives Gitter für Beispiel 3

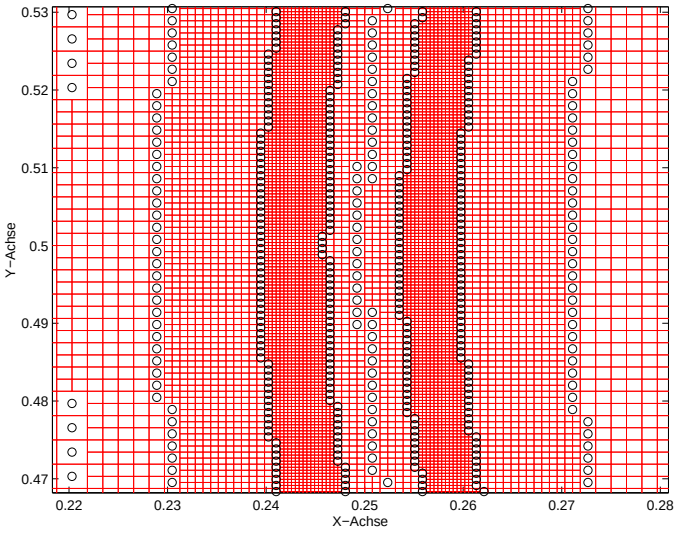


Abbildung 5.13: Zoom auf das Gitter für Beispiel 3

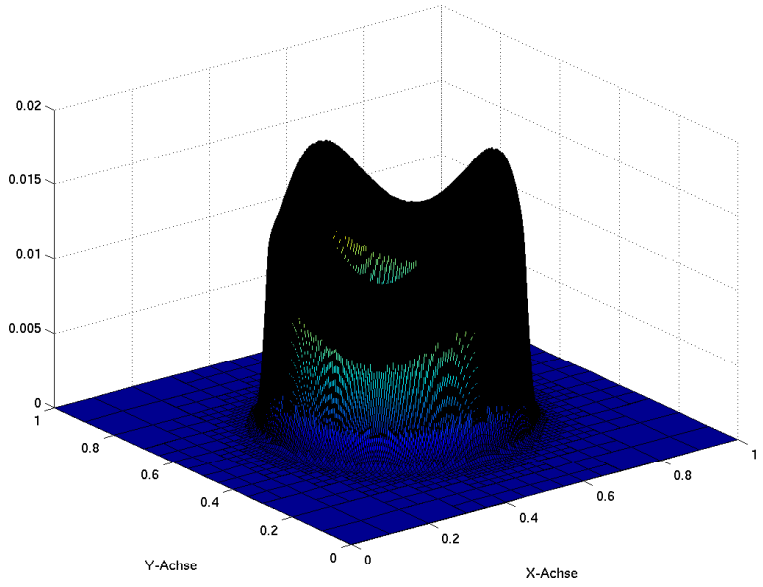


Abbildung 5.14: Fehler $|u - u_h|$ für Beispiel 3

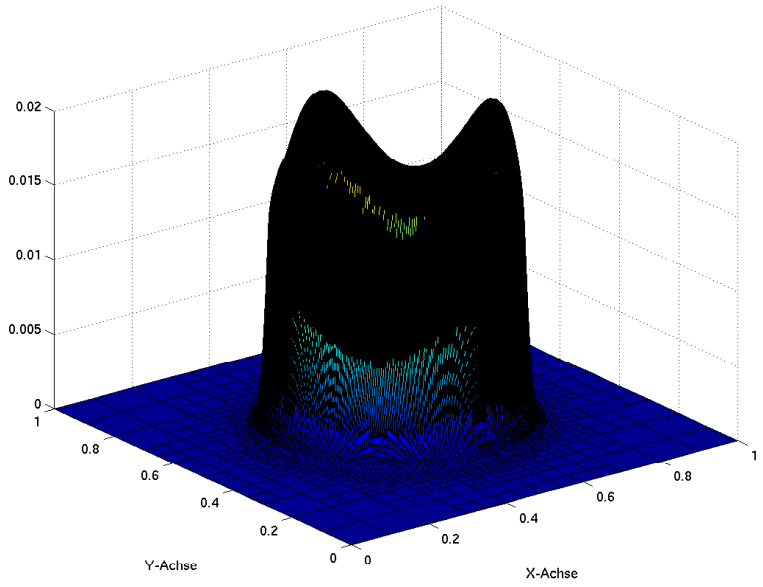


Abbildung 5.15: Fehler nach Postprocessing $|u - u_h|$ für Beispiel 3

Fazit

Im Rahmen dieser Arbeit wurde ein adaptives finite Differenzenverfahren für singulär gestörte Konvektions-Diffusions-Gleichungen in 2D entwickelt.

Dabei wurde ein gegebenes äquidistantes Gitter sukzessive verfeinert, bis die gewünschte Genauigkeit der Lösung erreicht werden konnte. Die hierbei verwendete Verfeinerungsstrategie wurde erläutert, wobei ein besonderer Schwerpunkt auf die Implementierung gesetzt wurde. Dieser Ansatz lieferte auch die Möglichkeit zur Parallelisierung, was aber im Rahmen der vorliegenden Arbeit nicht weiter verfolgt werden konnte.

Zur stabilen Diskretisierung der Konvektions-Diffusions-Gleichungen wurden upstream Differenzen 1. Ordnung verwendet. Um Interpolation zu vermeiden, wurde eine Koordinatentransformation eingeführt, welche die künstlichen Punkte mittels der Eckpunkte eines Quadrates diskretisierte.

Es wurde ein Fehlerschätzer auf der Basis eines gewichteten Residuums entwickelt. Dabei konnte gezeigt werden, das der Schätzer klein wird, wenn der Fehler klein wird. Es handelt sich also hierbei nicht um einen Schätzer im strengen Sinne, sondern mehr um einen Fehlerindikator. Die gute und verlässliche Funktionsweise dieses Indikators konnte an verschiedenen numerischen Beispielen überprüft werden. Auf dem adaptiv verfeinerten Gitter wurde ein Postprocessing entwickelt, welches für hinreichend weit verfeinerte Gitter gute Ergebnisse erzielen konnte.

Das gesamte Verfahren wurde an verschiedenen Beispielen getestet. Hierbei konnten durchweg gute Resultate erzielt werden. Sowohl im Hinblick auf die Genauigkeit der Lösung, als auch im Hinblick auf den benötigten Rechenaufwand. Das entwickelte Differenzenverfahren erwies sich als robust und wenig bis kaum parameterabhängig und erfüllt somit alle Voraussetzungen, um als

Blackbox-Löser dienen zu könnnen. Es ist durch die Verwendung nicht äquidistanter Schrittweiten auf Gebiete mit beliebigen Geometrien anwendbar und nicht auf einfache Gebiete beschränkt.

Interessante Weiterentwicklungen des hier vorgestellten Verfahrens zur Lösung von Konvektions-Diffusions Gleichungen sind u.a. die Erweiterung auf zeitabhängige Probleme, sowie die Verwendung bei der Lösung der Navier-Stokes-Gleichung. Eine Erweiterung auf 3D Probleme stellt ebenfalls eine wichtiges Aufgabengebiet dar. Dabei kann die verwendete Verfeinerungsstrategie jedoch nicht eins zu eins übernommen werden, sondern bedarf der Erweiterung um dabei auftretende Spezialfälle. Die Parallelisierung des Verfahrens - sowohl in 2D als auch in 3D - stellt ebenfalls ein noch offenes und interessantes Problem dar.

Literaturverzeichnis

[ADD96] P.R. Amestoy, T.A. Davis, I.S. Duff: *An approximate minimum degree ordering algorithm.*
SIAM J. Matrix Anal. Applic., 17(1996), 886-905

[Ag65] S. Agmon: *lectures on elliptic boundary value problems.*
D. van Nostrand company, INC. (1965)

[ADN59] S. Agmon, A. Douglis, N. Nirenberg: *Estimates near the boundary for solutions of elliptic partial differential equations satisfying general boundary conditions.*
Comm. Pure Appl. Math., 12 (1959), 623-727

[AN98] O. Axelsson, M. Nikolova: *Avoiding Slave Points in an Adaptive Refinement Procedure for Convection-Diffusion Problems in 2D.*
Computing, 61 (1998), 331-357

[Bo81] E. Bohl: *Finite Modelle gewöhnlicher Randwertaufgaben.*
Teubner, Stuttgart (1981)

[Da04] T.A. Davis: *A column pre-ordering strategy for the unsymmetric-pattern multifrontal method.*
ACM Trans. Math. Software, vol 30, no. 2, (2004), 165-195

[DGLN04] T.A. Davis, J.R. Gilbert, S.I. Larimore, E.G. Ng: *A column approximate minimum degree ordering algorithm.*
Math. Software, vol 30, no. 3, (2004), 353-376

[Gr97] A. Greenbaum: *Iterative Methods for solving Linear Systems.*
SIAM, Philadelphia (1997)

[Gu88] C. Günther: *Vergleich verschiedener Differenzenverfahren zur numerischen Lösung der Konvektions-Diffusionsgleichung.*
Bericht KfK 4439, Kernforschungszentrum Karlsruhe (1988)

[GS74] V.A. Gushchin, V.V. Shchennikov: *On a monotone difference scheme of the second order* (in Russian).
Z. Vycisl. Mat. i Mat. Fiz., 14 (1974), 789-792

[GT83] D. Gilbarg, N.S. Trudinger: *Elliptic partial differential equations of second order.*
Springer, Berlin (1983)

[GMS92] J.R. Gilbert, C. Moler, R. Schreiber: *Sparse matrices in MATLAB: design and implementation.*
SIAM J. Matrix Anal. Applic., 13 (1992), 333-356

[JRB95a] C. Johnson, R. Rannacher, M. Boman: *Numerics and Hydrodynamic Stability: Toward Error Control in Computational Fluid Dynamics.*
SIAM J. Numer. Anal. 32 (1995) 4, 1058-1079

[JRB95b] C. Johnson, R. Rannacher, M. Boman: *On transition to turbulence and error control in CFD.*
Preprint 95-06, Universität Heidelberg, 1995

[Ha86] W. Hackbusch: *Theorie und Numerik elliptischer Differentialgleichungen.*
Teubner (1986)

[KT78] R.B. Kellogg, A. Tsan: *Analysis of some difference approximations for a singularly pertubed problem without turning points.*
Math. Comp., 32 (1978), 1025-1039

[Le79] B.P. Leonard: *A survey of finite differences of opinion on numerical modelling of the incompressible diffusion convection equation.*
Finite elements methods for convection dominated flowd (ed.: T.J.R. Hughes). ASME New York (1979), 1-30

[Mi77] J. H. Michael: *A general theory for linear elliptic partial differential equations.*
J. Diff. Equations, 23 (1977), 1-29

[PW67] M. H. Protter, H. F. Weinberger: *Maximum principles in differential equations.*
Prentice-Hall, Englewood Cliffs (1967)

[RST96] H.-G. Roos, M. Stynes, L.Tobiska: *Numerical Methods for Singulary Perturbed Differential Equations Convection-Diffusion and Flow Problems.*
Springer-Verlag, (1996)

[St79] G. Stoyan: *Monotone difference schemes for convection diffusion problems.*
ZAMM 59 (1979), 361-372

[SW38] G.H. Shortley, R. Weller: *Numerical solution of Laplace's equation.*
J. Appl. Phys. 9 (1938), 334-348

[Ya81] M. Yannakakis: *Computing the minimum fill-in is NP-Complete.*
SIAM J. Alg. Disc. Meth., 2 (1981), 77-79